

On using the laptop in improvisation.

- Constrained interfaces, engaging instruments

Anders Tveit

Master's thesis submitted in partial fulfillment of the
requirements for the degree of Master of Arts in the
Department of Musicology, University of Oslo

Autumn 2018

Abstract

This thesis is concerned with using the laptop within the context of live improvised music. Specifically, developing a number of instruments which uses the laptop's built-in interfaces for control input and interaction. In the development of these instruments and the challenges that arise from using the laptop's constrained native physical interfaces for interaction I am investigating how they can be designed and used to facilitate an engaging activity in the interactive relationship in an improvised music context. An activity where I, as a performer, am able to focus, react to and interact with instant actions carried out in the musical environment. The work is informed by historical and theoretical studies, where important questions and reflections related to the laptop as an instrument, its historical development and the sonic and performative attention it gives to both the performer (s) and listeners are important relevant issues in the development of digital instrument.

Acknowledgements

It took a while.

I would like to acknowledge the contribution made towards the completion of this thesis by my supervisor Alexander Refsum Jønsens for encouragement and extreme patience.

To all of the incredible musicians that I have been fortunate to be able to play with, without you none of this could have happen. Special thanks to Terje Evensen for being my regular musical sparring partner over the years.

To the most important people, Helga Kristin, Levi and Marie:

Now it's done

Tønsberg, November 2018

Anders Tveit

Contents

1 Introduction	4
1.1 Research Questions	6
1.2 Overview of structure	7
1.3 Premises	8
2 Historical background	10
2.1 From the studio to the stage: Live electronic music	11
2.2 Computers, music and performance -Analog to digital, the in-between	14
2.3 Digital-digital, MIDI and interactivity	16
2.4 The emergence of real-time digital signal processing	23
2.5 Towards the laptop era	25
2.6 Wearing - Off	29
2.7 New approaches -from live coding to laptop orchestras	30
2.8 Post-laptop?	32
2.9 Summarising and brief discussion	34
3 Human computer interaction and digital music instrument design	37
3.1 On human computer interaction and digital instruments	37
3.2 The conceptual relationship, constraints and affordances	39
4 Laptop Instruments	43
4.1 The keyboard variations	44
4.1.1 $\alpha\beta$	45

4.1.2 $\alpha\beta$ _Redux	47
4.1.3 $\alpha\beta$ _Polypoly	49
4.2 Summary of the Keyboard variations	51
4.3 Multi Touch	52
4.3.1 at-multitouch	53
4.3.2 at-crossConv+Sampler=Continuum?	55
4.3.3 at-granular	58
4.3.4 Other variations	61
4.4 Summary of Multi-Touch instruments	62
4.5 Lights, Camera, Action?	64
4.5.1 The Smack Machine	65
5 Evaluation	67
5.1 Performances, conditions, and the field	67
5.2 On Evaluation	69
5.3 The Multi-Touch Instruments	70
5.4 The keyboard variations	75
5.5 The Smack Machine	78
5.6 Remarks on the evaluation	79
6 Concluding remarks	81
Appendices	90
A List of files included digitally	91

Chapter 1

1 Introduction

In this thesis I present an approach for using the laptop in live improvisation. More Specifically, developing a number of instruments which uses the laptop's built-in interfaces for control input and interaction. In the development of these instruments and the challenges that arise from using the laptop's constrained native physical interfaces for interaction I am investigating how they can be designed and used to promote an engaging activity in the interactive relationship in an improvised music context. An activity where I, as a performer, am able to focus, react to and interact with instant actions carried out in the musical environment.

The laptop computer has a number of embedded or native input peripherals such as the keyboard or trackpad. Sensors like accelerometers, light-sensors (used for adjusting the lcdscreenlight according to ambient lighting) hi-resolution cameras and multitouch technology is often found in many of todays laptops. These are input devices which through some software «hacks» and creative thinking can be used for musical expression. Using the laptop's native input capabilities for musical expression have already been creatively explored by a relatively new music-phenomena; the laptop orchestra. Examples of such orchestras are : The Princeton Laptop Orchestra, The Stanford Laptop Orchestra, Huddersfield (HOL), Oslo Laptop Orchestra (OLO) and many more. But to my knowledge, little has been written about the experience and artistical exploration of using this in improvisation and performance outside a laptop orchestra.

Performing improvised music with the use of electronics and the computer, has for the past 15 years been a part of my musical expression as a musician. In the course of these years, a vast number of interaction schemes and external MIDI hardware controllers were tried out, using the computer in the context of improvised music. Some of them were better than others, but the «ideal-controller-of-the-day» constantly shifted with the current trends and the nature of the material, parameters and so on I wanted to control, in conjunction with the preferred sonic result. A frustrating and perhaps a familiar situation for many computer music artists. I then became more and more interested in limiting myself to fewer hardware controllers, to fewer sliders and less "GUI knob fiddling". Reducing this knob fiddling or studio-centric approach and the cognitive overload this so often led to, was essential in order to try to have a better and more direct sonic interaction with my fellow instrumental acoustic and electronic performers. Taking a constrained and perhaps a dogmatic approach, eventually led me to experimenting with the use of the laptop itself as an approach for interaction and control. Inspired by typically "everyday use" of laptop interaction (typing the keyboard, pressing, pinching, sliding the trackpad etc.) I forced myself to think "outside the box" both for developing instruments with constrained interaction and conceptual action-sound relationship. This thesis is very much documenting this journey. I identify not only my experiences in the work as a performer, but will hopefully also shed some light on challenges and present strategies and solutions that might be applicable to the improvising musician who uses the laptop as a musical instrument.

1.1 Research Question

"Sounds are placed: placed in contrast to, in parallel to, in imitation of, in respect of, without regard to, other sounds. Minds struggle, coalesce, defer or acquiesce. Inner debate meets outer debate. Instant decisions dictate the immediate direction of the music." Eddie Prévost *No Sound is Innocent* (Prévost 1995:3)

The quote by Eddie Prévost is good description of the multifaceted, intricate and engaging activity of improvisation, a focus of the activity itself. One of the problems using the computer in improvisation is often the lack of immediacy and the non-intuitive approaches of the various sound producing software. So not being to be able to react and interact with your own instrument and other performers instantaneously I believe, are hindering the importance of an engaging activity, and ultimately hindering the instant decisions that dictate the immediate direction of the music.

By using the laptop's constrained native physical interfaces for interaction my research question becomes: How can these laptop instruments be designed and used to facilitate an engaging activity in the interactive relationship in an improvised music context?

To explore this question there are also subquestions that follows:

What are the issues and aspects of using the computer as a performance instrument ?

What are important specifications for developing Laptop based instruments for improvisation? How can I evaluate the laptop instruments in relation to my topic?

These questions will need to be addressed through both theoretical and practical work. The practical work in this project does not only consist of the technical development but also artistic work of using instruments. I see this as highly important in order to gain an understanding about the instruments behaviour and therefore reveal important issues that could not be assessed otherwise.

1.2 Overview of the structure

This Master's project is divided into two main parts; this written thesis and a practical part, where the latter will be referred to as modelling. The modelling is also split into two parts:

1. Software development: Design and development of the instruments.
2. The artistic work: Performance and recorded documentation.

The instruments, audio and video examples from this work is included with thesis as digital files. The complete list of files is listed in Appendix A.

The written thesis:

Provides the theoretical and documentation background to the project.

Chapters 2 and 3 is providing the theory with historical, technological and other aspects related to the design and the use of computer in music performance.

Chapters 4 and 5 presents the documentation of the modelling part with discussion and evaluation of experiences from the software development and artistic work.

Chapter 6 Concluding remarks.

The project structure with the written thesis and its relationship to the Modelling as illustrated by Figure 1.2.

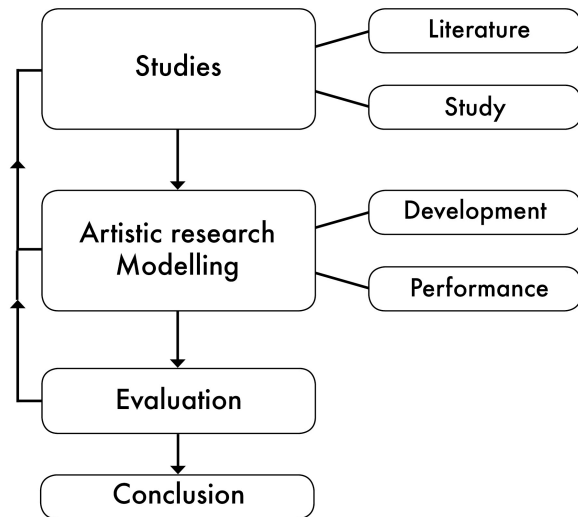


Figure 1.2. The structure of the thesis and its relationship.

1.3 Premises.

I would like to point out that this thesis is based upon material from many different disciplines such as music-performance, musicology, psychology, cognitive science, computer science and human– computer interaction (HCI) And it is outside the scope of this thesis to give the reader a in-depth insight into these disciplines, but I would like to encourage the curious reader to further read the sources I provide in this thesis. That being said, it also should be noted that there are cases where the different disciplines sometimes use slightly different definitions. In such cases I will therefore keep a simple approach and use common definitions in relevance to the topic and not discuss this in depth, but use footnotes where I find it necessary. This thesis will not address in any detail to research in sound synthesis and DSP (digital signal processing) other than referring to some techniques and appliances briefly described in the following chapters. Throughout this thesis I will refer to various ways of

using the computer in music performance, The term digital musical instrument (DMI) is adopted in this thesis for describing a general system consisting of a controller, the sound-engine and the mapping between these (Miranda, Wanderley 2006). But in order to clarify when referring specifically to the use of the laptop as controller, sound-engine and the mapping in its entirety, I have chosen to use the term Laptop Instrument. Although both terms use the word instrument with its connotations to traditional instruments, there is as I will address in Chapter 2 many examples of using the computer in performance that don't fall so clearly into this paradigm of one action to one sound event. I should also point out that I opted to use the word *action* instead of gesture, as action describes a coherent and goal-directed movement (Jensenius 2007). In order to prevent confusion around the use of the terms real-time and live as they don't necessarily mean the same thing. Real-time is in this thesis used in relation to computer processing time. This can be in respect to an audio DSP process where the input and output is processed continuously in the time it takes to in- and output the same samples independent of the processing delay. Without going in to any deeper definition of the term live, the term is simply used in this thesis to point out a human-musical action done and perceived in human-time.

Chapter 2

This chapter presents a section with a historical background of using computers in performance and improvisation. Followed by a section with a brief presentation of current, related work and research relevant to this thesis. Starting with the analog Live electronics works via MIDI interactive and improvisation systems followed by laptopism , live-coding and Laptop Orchestras and beyond.

2 Historical background

The advances in computing power and software along with decreasing costs and democratisation of technology during the last decades have not only pushed the boundaries for digital sound generation and manipulation. But the possibilities of doing so live with real-time processing capabilities has greatly affected the use of the computer in production, composition and performance. In our post-digital world the computer is ubiquitous in music, ranging from powerful composition and music production tools to performance instruments. Although the core use of computers in music is rooted very much in composition and music production, It has throughout the history of computer and electronic music found its ways used in improvisation and as performance instrument in different forms. Although the focus in this chapter is on using the computer as a performance instrument rather than a composition tool, with electronics and computers in the performance loop as we will see, the borderline is sometimes not that easy to identify or draw between composition, improvisation, instrument and the performer. Emmerson (Emmerson 2007: 115-16) designates three aesthetic paradigms in relation to the notion of "live"performance activity and the historical development in technology.

1. Analog electronic live signal processing of any source.

2. Personal Computers and "real-time" control and event processing via the MIDI standard.
 3. Emergence of laptop computers and real-time digital signal processing of any source.
- In addition to these paradigms I will add the future possibility for identifying a fourth one;
4. The ubiquitous computer and mobile music making platforms.

In the following sections I will highlight issues and aspects of using the computer as a performance instrument by looking closer at these paradigms and the historical background, from early live electronic music, the MIDI era, leading up to the present days use of the laptop in performance and improvisation and beyond. And at the end provide summarisation and discussion on how the computer today encompasses both the studio and performance systems. There are of course many aspects and historical events that is not covered in this chapter for the simple fact that it is out the scope to provide a complete overall view.

2.1 From the studio to the stage: Live electronic music

During the post war period after the tape recording technology was invented, composers was now provided with a technology that made it possible to compose treating any sound source as compositional resource. The new equipment and technology in the sound studios afforded the composers a precise control of every detail of the sound. By recording the result to tape, into fixed compositions such as in the music concrete and electronic music traditions the need for a live performer or instrument was removed. The playback of the tape recording was the performance of the composition itself. A performance or realisation of such a piece, the sound speakers acted as the "performer" as in the acousmatic tradition or combining the tape playback with live performed instruments. Early specialised sound equipment and synthesisers were massive, expensive and often inaccessible as they were located in specialised sound studios or research centers. Although even earlier electronic instruments such as the Theremin (1920) and Ondes Martenot (1928) had been advocated as modern electronic instruments "that everybody could play" They were in-fact notoriously difficult to

play and "one-dimensional" sounding (Collins 2007). As electronic analog instruments, effects and sound equipment in the late-1960s started to get more affordable and obtainable they were also starting to be explored as new performance instruments by a range of experimental composers and performers in Europe, America and Japan. In opposition to the highly composer controlled, studio-based and pre-recorded tape compositions, this approach dubbed Live electronic music was to explore spontaneity, dialogue, discovery and group interaction (Sutherland 1994). Not willing to confine to a "one-sounding" electronic instrument such as the theremin which itself had not changed the nature of musical performance or to use any pre-recorded sounds from tape. The idea was to take elements of studio-based composing and make it novel performance instruments (Collins 2007). Using a range of different circuits and sound equipment and by not having a sound itself but an external input; any sound source picked up from a microphone or an electronic signal could be amplified and transformed through filtering, tape-delays, modulation effects and other devices. By taking the input, one could transform a sound and thereby constructing a new sound on the spot and further manipulate it during the course of an performance. This added an important aspect of new aesthetic considerations to electronic music, the element of interaction and improvisation. The strict studio-based procedure of composing, treating any sound source as compositional resource shifted to an performance based activity, treating any sound source as a resource for interaction and live improvisation. During the late 1960s and early 1970s different live electronic ensembles and collectives were formed focusing on live improvisation such as the: *Gruppo di Improvvisazione Nuova Consonanza*, *Group Ongaku* and *MEV, Musica Elettronica Viva* and *AMM* (Manning 2004). As the electronic devices, singularly or in various combinations and connections made up the performance instrument itself, what became an important element was not only improvising and performing using electronic devices. But developing and designing devices and the signal chain was itself an important activity, creating personal instruments. In North-America, inspired by the early electronic work of John Cage¹; composers and performers like David Tudor, and members of

1. Most notably: *Imaginary Landscape No.I, II and III* (1939-42) and *Cartridge Music* (1960) which featured turntables, amplified wire coil, contact microphones and Sine oscillators.

the collective Sonic Arts Union had a pragmatic "Do-It-Yourself" attitude to developing their own circuit designs and signal chains. As Nicholas Collins comments in his article "Live electronic music" (Collins 2007: 46):

"David Tudor saw this process of designing these instruments and their sound, as a way that new musical material and form would emerge as each instrument took shape" (ibid).

David Behrman and Gordon Mumma, members of the Sonic Arts Union constructed several of their own musical circuits that was relatively simple but in combinations could form complex and rich textures from even simple inputs. Gordon Mumma in-particular made performance circuits by combining pitch detection mechanisms, gating, feedback processes and envelope followers that could respond actively to signals during a live performance, or self-adjust to the acoustic properties in the performance space (Holmes 2008). Leaving him free to focus on the higher level control aspects of his instruments. The circuits by Behrman and Mumma foreshadowed the interactive computer music to come in next decades with the advent of the MIDI standard in the 1980s. But in contrast to computers, their analogue devices and circuits was not dependent on any memory or processing delay, everything was done "in-a-now", a performance existed as a function of human awareness passing in time. Through this Live electronics approach not only is the studio procedure transferred to the stage, but the distinctions between composition, improvisation activities and the instrument is blurred and fused together into one thing. An aspect which also becomes important to note in today's use of computers in music performance where real-time live processing of sound and programming is a way of 'doing electronics in software, something which I will shed some light to later on in this chapter.

2.2 Computers, music and performance -Analog to digital, the in-between

Before the advent of microcomputers and high level software in the 1970s, The only possible way of making music using a digital computer at the time was at universities or computing centers. These huge mainframe computers originally designed for science and data collection where cumbersome to program using punch cards, required special expertise and the musical result was limited. The computational power of the mainframe computers was also very limited, to even produce a minute of music could potentially take hours or days (Manning 2004). Although the pioneering work of Max Mathews at Bell Laboratories and his MUSIC N² series of programs was both revolutionary and promising, the sheer physical size, costs and processing time of the computers made it impossible to be used in performance. But with the introduction of digital integrated circuits and the microprocessor in the early 1970s the computer became increasable faster and the memory was hugely expanded. The size and costs was also greatly reduced, into what was now effectively called mini and microcomputers. It was still not possible to do digital synthesis or any digital audio processing in real-time, but Max Mathews had worked on GROOVE; a hybrid system that combined a digital computer controlling an analog synth. Calling on the enhanced control capabilities of the computer one was provided with the possibility to interact more directly with processes of sound. Especially since the GROOVE system provided a graphical user interface where the parameters were visually updated via the computer screen and could be controlled with the typing keyboard and range of joysticks. Max Mathews viewed his system as an instrument where the composer first worked with traditional programming techniques leaving interactive modifications to be done during a performance. Although promising there

-
2. MUSIC-N refers to the family of the programs called MUSIC. MUSIC I was developed in 1957 at Bell Laboratories. A much later version, MUSIC 11 written by Barry Vercoe was translated to the C programming language in 1986 and become the forerunner to Csound.

was little usage of this system in performances as it was mainly used for research (Manning 2004). In 1975 the KIM-1 single board microcomputer, a predecessor of the first Apple computer was released to the commercial market. And the San Francisco based collective "The League of Automatic Music Composers" led by Jim Horton is regarded as the first musicians to incorporate these newly available microcomputers in live musical performance. As early as 1977/78 this improvising collective used the computers to mostly control their own network of analog or digital sound-making circuitry instruments and effects. John Bischoff, one of the founding members comments on their aesthetic process:

" The music was always live, with no sequences pre-planned. Each player's "station" played its own composition, had its own sound-making equipment, and would send and receive information to and from the others. The meaning of this information might be completely different on one end of the exchange and the other: a pitch indication from one player might be controlling the rhythm of the other; for example.....For us, the music was never "in the computer." The microcomputers were always just components with particularly interesting behaviour to incorporate into our networks which included other electronic circuitry, as well as human beings. The heart of the work was in physical bricolage or assemblage, an essentially sculptural musical practice " (Perkis, Bischoff 2007) .

Members of The League of Automatic Music Composers refined this network computer system, and later on become "The Hub" regarded as an early forerunner to today's laptop orchestras. Other interesting live performance work done with the microcomputers was by the aforementioned Nicolas Collins in his improvisational piece Little Spiders from 1981³ where two microcomputers is used to examine the interaction and gestural styles of two

3. Nicolas Collins, *Little Spiders* (1981) on Nicolas Collins & Ron Kuivila, *Going Out With Slow Smoke*, Lovely Music LP, 1982.

keyboard players, producing sound based on the individual's actions⁴ (Collins 2009). This transition or in-between period going from analog to digital expands further on the early interactive music ideas set in motion in the late 1960s; exploring group interaction and adding the computer as a part of this loop. As the microcomputers became more powerful and standardized the emphasis shifted to gradually to the use of software. Composers or performers who had previously learned electronic circuitry out of necessity, soon found themselves as being composer/programmers as well. Programming computer-code is a linear process where instructions to the computer is done sequential, this also forced the artists to think differently about their organisation, variation and playback of music. Where electronic circuits are transitory and vertical in conception. Software is horizontal, stretched out in time (Holmes 2008).

2.3 Digital-digital, MIDI and interactivity

Parallel to the experimental live performance work done by artists such as "The League of Automatic Music Composers" and others, groundbreaking work on digital real-time synthesis, analog-to-digital conversion and music software had now been done at research centers by Max Mathews, Xavier Rodet, Jean-Claude Risset, John Chowning and others. Although early on limited to academic studio composers and researchers, the technology found its way by the early 1980s into the new digital synthesizers such as the revolutionary DX-7 by the Yamaha corporation.⁵ Or the earlier but extremely expensive computer music system Synclavier I by the New England Digital Corp (\$25,000 to \$200,000). And later on to sound chipsets based on the new synths to computers which by now, just in a few years time had gotten even smaller and faster than the microcomputers of the late 70s. Software applications for music was released to the public and with the MIDI (Music Instrument

4. Score/Performance directions from Nicolas Collins: www.nicolascollins.com/texts/spidersscore.pdf

5. The Yamaha corp. licensed the FM synthesis by John Chowning in 1975 and implemented it into the DX-7 digital synth released in 1983.

Digital Interface) standard in place in 1984 one was provided with a standard interface connection that computers for-instance could control a digital synthesiser. Around the same time the first commercial digital samplers such as the famous Fairlight CMI (Computer Music Instrument) was introduced and revolutionized the whole music industry. Digital samplers such as the Fairlight CMI was basically a powerful computer system (at the time) with analog-digital-analog converters, a keyboard controller, a computer screen, ASCII keyboard, and graphical editing and sequencing software. Albeit at a huge cost (\$25000 in 1979) The Fairlight CMI had with the analog-digital converters the possibility to record any sound, analyse and store it on the computer, manipulate it via the software and then play it back using the keyboard or a real-time graphical pattern sequencer, which for many music producers and artists was the selling point. Being able to play a pre-recorded sound on a keyboard was something that was not entirely new in itself. The Chamberlin and the Mellotron, as famously used by the Beatles and other popular artist had been in use in live performances since the 1960s. These tape-based "samplers" was essentially electro-mechanical keyboards with the possibility to playback polyphonically short loops recorded on tapes connected to each of the keys. But the number of sounds was limited with three to six factory recorded sounds, depending on the model. To use your own recorded sounds wasn't impossible, but generally involved sending the recorded material to the Mellotron factory to be converted into a rack of tapes for the specific model. With the computing power of a computer as in the Fairlight CMI the possibility to record any source material, manipulate, relatively quickly store and recall it, there was both a creative and a performance advantage over the Mellotron and Chamberlin. As the analog synthesizer was being replaced with the power of digital synthesis, similarly the tape recorder and the mellotron was being replaced with the digital sampler. The MIDI standard which had become available on personal computers, samplers and synthesisers around 1984 proved not only to be an effective standard (not without limitations) but also led to a paradigm shift in music and performance. MIDI have two main purposes: First to provide a standard hardware interface connection in order to facilitate communication between computers, samplers or synthesisers. Secondly, this communication is standardised into a data encoding scheme for event processing of musical performance and control data. The concept of an event in the MIDI

specification is referring to its "notes on/off" definition and reflected changes in the parameters: pitch and velocity sent to the sound-producing unit along with control parameters such as: vibrato, panning, tempo and so on. Since the amount of data required of transmitting and receiving was small enough at a standardised serial transfer rate, (@ 31250 baud) MIDI event processing could be done in "real-time" with the available computers of the time.⁶ Although there is several cases where MIDI fall out of what can be considered "in real-time". An increase in polyphony and control parameters, can lead to a so called "MIDI choke" where the data flow is severely delayed, resulting in latencies well beyond 50ms. (Manning 2004). But despite some of it's flaws, with MIDI there was now a very clear and standardised separation between the input (control) and the output (sound) as opposed to the traditional instrumental scheme, where there is no clear separation between the playing interface and its sound generating sub-systems (Jorda 2007). With MIDI any input parameter from the control interface, be it a keyboard or computer software could be assigned to any sound or parameters on a digital synthesiser in a standardised way. Anything inputted can be connected to output, only limited by the imagination of the performer/developer. This gives the performer a freedom to not only have a one-action to one-sound event as in traditional instruments but also have one action-to-several sound-events or even to larger musical structures. The expense of this freedom is that there are not necessarily any direct or clear relationship between cause and effect. The challenge with computers, synthesisers and such parametric control in performance; was that being able to think fast enough to specify controls and quickly changing variables in a live performance was a daunting task. One of the approaches to address this issue was by integrating the computer and the performer in an tighter interactive relationship by having shared control. Composer and performer Joel Chadabe had identified these issues early on. In 1977 he worked with a concept where one could automate a musical process but then interact with that process while it was running. His analogy to this was the fly-by-wire system as in airplanes; pilots would tell the computer what the plane wants to do and the computer flies the plane (Holmes 2008).

6. Real-time event processing should not be confused with real-time digital signal processing, as this still was not possible with personal computers at the time.

Chadabe coined his concept interactive composition, by letting a computer program having the control of the vast number of sound producing parameters, the performer would then interact or "piloting" through higher level controls. Chadabe was one of the earliest user of Computers and digital synthesiser systems in live performance and in his piece "Solo" from 1978 he showcases his interactive concept with the use of the first Synclavier computer-and-digital-synthesiser. In "Solo" the computer system plays eight melodies and it's accompaniment chords on various FM synthesised sounds (clarinets, flutes and vibraphone) With the aid of two theremins solely used as controllers, he controls the duration of the notes with one theremin and the relative volume with the second, thereby affecting the overall timbre. As he could not foresee which chords for example the clarinets would play, the chords they produced determined his decisions for the next musical event produced. (Chadabe 1997) In "Rhythms" (1980) and "Follow Me Softly" (1984) Chadabe also explored the computer keyboard as an interface with the same Synclavier system. Although linked to improvisation, Chadabe pointed out that interactive composition (as a process) was something different from both improvisation and composing:

[interactive composition] "... This is different from typical instrumental improvisation in that an improviser specifies and performs data in reaction to another improviser, but here the composer specifies and performs control strategies rather than data...The difference between instrumental improvisation and control strategy interaction is, then, primarily that of one's position in a control hierarchy, but it is true that in both cases the performer must act and react in realtime". (Chadabe 1977: 7)

Others who explored interactivity and computers in music performance was accomplished jazz musician and scholar George Lewis. Where Joel Chadabe's interactive compositions relied on a human performer Lewis took another perspective of interaction with the computer by using it autonomously, as a second performance partner. In his personal interactive improvisation system the "Voyager" The computer was programmed to "listen" to what he played on the trombone by tracking the pitch of the trombone and reacting and improvise via MIDI to what was being played. Another example of interaction and shared control was the

"computer network band" The Hub. Founded in 1985 and consisting of former members of the aforementioned The League of Automatic Music Composers (LAMC). The Hub was in many ways the LAMC's successor by refining and further develop some of the computerised network ideas. The Hub's members used their own computer setup for generating sound, and each player wrote a computer program which made musical decisions in response to messages from the other computers in the network along with the control actions from player himself. Central to the band was a centralised computer system called the The Hub, this computer was used to pass messages between the members, serve as a common memory and keep information about each players activity that was accessible to the other players' computers (The Hub 1989). Depending on the rules of interaction, the music could then range from free improvisation to to tightly synchronised performance. The rules or framework in the improvisational piece "The Glass Hand" required each performer to come up with a set of predetermined sounds; devise a method of segueing from one sound to the next by smooth transition. Where the triggering and speed of their transitions was based on information received from other players and on the characteristics of their current sound. Each performer was responsible for sending a triggered message to one other player (b), and a speed message to one other player (c) in the group. When a trigger message was received, the player was supposed to begin a transition from the current to a new musical state, using a transition rate determined by the value of the speed message they received (Perkis in Dean 2003 :114). The Hub later on replaced their central network computer with the Opcode Studio V MIDI interface as "the hub" but acknowledged that MIDI's note-oriented nature was in certain ways inappropriate for their uses (ibid). Coinciding with the MIDI era and the burgeon field of human-computer interaction research; affordable and fast 16 bit personal computers such as the Amiga, Atari ST and Apple MacintoshPlus was introduced to the public. These computers where GUI-centric offering high resolution interactive graphics with the new WIMP (window, icon, menu, pointing device) environment and tailored as multimedia machines. Among the audio features they had where built-in sound chips capable of simple real-time 8-bit digital synthesis in 2-4 channels and digital sample playback. MIDI and Sampling (analog-digital converters) capabilities where either built-in (as the MIDI features of the Atari ST) or available as hardware extensions. As the need for custom hardware and

expensive specialised computers was now greatly reduced, software was the new "untapped" source. Development of the graphical user interface and emergence of the music software industry in the mid '80s removed the need for composers and performers to have highly trained skills in solely text-based computer programming. Although most of the software where focused on composition tools such as MIDI sequencing and score notation, there where some that where more performance oriented. The programming language HMSL (Hierarchical Music Specification Language) 1980-1996 by Larry Polanski, David Rosenboom and Phil Burk were aimed at musicians and composers that worked with MIDI and live performance. HMSL was available for Apple Macintosh and Amiga and included tools for exploring algorithmic composition, intelligent real time instrument design, MIDI control and response, musical cognition and perception. (HMSL). In 1985/86 musician, composer and programmer Laurie Spiegel developed a software she billed as a "intelligent instrument" originally made for the Apple Macintosh 512 and later ported to the Amiga and Atari computers. The software "Music Mouse" utilised the computer mouse as the main controller and derives 4 voice harmony from the 2-dimensional mouse movement while controlling a range of available musical parameters from the computer keyboard. Spiegel embedded automated processes such as harmonic rules into the program based on the selected pitches from the mouse movement (Spiegel 1987). The "Music Mouse" was not alone into early ventures using the computer as an instrument, "Instant Music" by Robert Campbell and Electronic Arts for the Commodore Amiga 500 (1986) had similar features as "Music Mouse" but where simpler and marketed more as a musical game. "Instant Music" was "played" using the mouse movement in one dimension (y), producing pitches in real time which where subsets of predetermined chord progressions of music pieces that came with the program (ibid). Sonix by Aegis (1986) for the Amiga and Deluxe Music Construction Set (1986) for the Apple Macintosh and Amiga was originally conceived as composition and MIDI production tools, but featured 8 bit digital sample playback and simple real-time digital synthesis which could be played on the computer keyboard. "Sonix" also had its own graphical sound-editor where you draw your own waveforms and manipulate the parameters of the synth sounds in real-time by using the mouse.⁷ Resembling the virtual software synths

7. [http://elisoftware.org/index.php?title=Sonix_\(Amiga,_3_1/2%22_Disk\)_Aegis_Development_-](http://elisoftware.org/index.php?title=Sonix_(Amiga,_3_1/2%22_Disk)_Aegis_Development_-)

in coming decades. In 1986 inspired by the GROOVE project and Conductor programs by Max Mathews as well as the experiences from his own performance systems Joel Chadabe established his software company Intelligent Computer Music System. With the goal to enter the commercial market with interactive music software for personal computers. Together with programmer David Zicarelli they created the interactive music software "M" released in 1987. As it read on the front page of the manual:

"M is an interactive composing and performing system that take notes and chords that you specify and manipulates them, under your control, to create musical compositions which unfold during live performance" (Zicarelli 1988).

As the growing importance of interactive tools and a flurry of GUI interaction oriented software for personal computers were released in the mid-to-last 80s. Composers and researchers at computer music centers like the Paris-based IRCAM were becoming aware of the personal computers revolution and sought out to replace the text style user interfaces of the specialised computers and workstations with the more versatile and interactive environment available with personal computers. This marked a shift in orientation about computer music research and led to tensions between they who saw computer music as a performance art and those who saw it as a pre-compositional process (Manning 2004). In 1986/87 programmer and researcher Miller Puckette who worked at IRCAM at the time, took use of the GUI interaction and WIMP paradigm available for the Apple Macintosh.

"The Macintosh in question was brought into IRCAM by David Wessel, without whose efforts I and the rest of IRCAM might have entirely missed out on the personal computer revolution". (Puckette 2002: 34)

Puckette developed a graphical environment "The Patcher" to be run on the Macintosh for the

_1986_USA,_Canada_Release accessed 13/2 2012(Aegis 1986)

computer music programming language Max (named as a homage to Max Matthews). The Patcher in conjunction with Max was developed as way to give composers access to a system for interactive computer music and the Sogitec 4x (later ISPW/NeXT) computer workstation at IRCAM⁸ (Puckette 1988,1991, 2002). In 1989 Puckette and IRCAM licensed The Patcher/Max to the software company Opcode Systems, where it was re-engineered by David Zicarelli and named Max/Opcode. With the public available and widely distributed version of Max/Opcode, the whole computer music field where transformed (Dean 2003). Max/Opcode provided the possibility for musicians with limited programming ability using their personal computers to produce their own software for exploring Interactive music, design computer instruments or virtually for anything that worked with MIDI. With the standardisation and separation of control with MIDI, the power of digital synthesiser and samplers; and its begun incorporation into personal computers together with the interactive software explosion; One thing remained as the final frontier for music performance using computers: Real-time digital audio processing and manipulation.

2.4 The emergence of real-time digital signal processing

In the early 1990s the rapidly growing MIDI software industry and the ever-increasing power of personal computers where now closing-in in performance to powerful hardware synthesisers and samplers (Manning 2004). With cost reductions and advancement in DSP technology, dedicated DSP co-processors used for real-time audio where oriented towards the personal computer market as retrofit boards. As a result, software synthesisers, effects and digital audio production tools where now becoming integrated in MIDI sequencers. The crave for real-time DSP capabilities in personal computers led computer music institutions like IRCAM and others to realize the importance of personal computers and had started to release their digital audio and synthesis software for DSP enhanced personal computers and consumer computer workstations. This provided performers like Lawrence Casserley and his

8. The Macintosh where used as a control computer with the 4x/ISPW workstations as the audio processor.

seminal work with saxophonist Evan Parker a platform for exploring real-time audio processing in improvisation (Casserley 2001). In 1990 Barry Vercoe presented a version of Csound, the descendant of MUSIC-N with real-time audio processing capabilities for the DEC, SUNsparc and NeXT workstations (Vercoe, Ellis 1990). Commercial powerful real-time DSP systems like the M.A.R.S system by IRIS (Di Giugno, Armand 1991) and Kyma by Symbolic Sound Corporation (Scarletti 1990, 2002) combined their own visual programming language with dedicated multiprocessor DSP hardware solutions to be used with personal computers. Used by a range of composers and performers for live electronic works and computer music improvisations, with luminaries such as Luciano Berio, Karlheinz Stockhausen, Joel Chadabe and Led Zeppelin bass player John Paul Jones as early advocates of using these systems in live performance. Soon, by the mid-end of the 90s the standard CPU processor and memory capability on personal computers was powerful enough for doing real-time digital audio and reduced the need for DSP co-processors. Nearly a standard fitted on every computer where both analog-digital and digital-analog converters and upscaled or high-end converters was available as replacement for professional use (Manning 2004). Computers had now become an essential part for many musicians and composers in their personal studios, for doing digital audio recording, editing, synthesis and MIDI or as part in live performance. The explosion of popular electronic music genres such as Electronic Dance Music (EDM) and its sub-genres such as Techno, Jungle, DnB and so on was solely concentrated around the use of digital music technology and pushed the demands for new software products in the commercial market. In 1996 software company Steinberg released their VST (Virtual Studio Technology) digital signal processing software component as part of their recording/sequencer software Steinberg Cubase 3.02 and released the VST as a software development kit for use with other audio applications the same year (Steinberg GmbH 1996). This software component or Plug-in provided the host application, with additional functionality with real-time digital effect processing and synthesis. And a new music software industry was spawned around this technology with hundreds of both commercial and share/freeware developers. In 1996 James McCarthy released the powerful Supercollider programming environment for real-time audio and synthesis. The german software company Native Instruments released Generator 0.96 a digital modular synthesiser

for PC, the forerunner to their Reaktor series. The same year Miller Puckette also developed the Max derivate PureData (Pd) with real-time DSP functionality (Puckette 2002). And in 1997 David Zicarelli who had taken over the sale, distribution and the development of Max via his company Cycling74 also introduced real-time DSP functionality to Max. Max/MSP as it now was called, soon became one of the leading programming platform for interactive music and computer music performance (Dean 2003).

2.5 Towards the laptop era

The combination of the laptop's distinguishing feature: portability -designed to travel with the user and the ability to handle real-time audio processing along with the exchange of information and tools via the internet and the increasing availability of dedicated software; marked a paradigm shift in computer music performance and its practice.

" "Computer music," after all, once meant hauling computer towers onstage, something reserved these days for a select few." (Kirn 2011). Using desktop personal computers for music in the 90s meant working primarily in a fixed environment with the notable exceptions of the few artist and composers such as the aforementioned and others who hauled their computers on stage. Although portable computer solutions or laptop computers existed and had been in the market since the early 80s, laptop computers was conceived to function primarily as a mobile office, not capable of handling cpu intensive task such as real-time audio processing; so their performance at the time was not on par compared with the desktop computers. It was not until the late 90,-early 2000s miniaturisation and the increase in computing performance the laptop was rendered powerful enough to be considered as a desktop computer replacement.

"While desktop computers allowed us to customize our virtual work environment, the laptop's mobility has allowed us to customize our physical work environment." (Cascone 2003:1)

For artists based in popular electronic dance music (EDM) and its sub-cultures; who worked primarily with computers for recording and music production, the popularity and portability of the laptop opened up a series of possibilities for music that sends it beyond spatial anchorages such as the studio. In their computer based studios EDM artists used a plethora of software synths, sequencers, samplers and effects with extended functionalities beyond their hardware counterparts for creating their music. Despite this and the computer workflow efficiency advantage, artists often relied on using a setup of hardware samplers, synths and effects in live performance as they were simply more "stage-ready" and portable than dragging along a desktop computer with all its peripherals. The modular approach typical for the genre(s) is to create a new work or a re-interpretation (remix) from pre-recorded or pre-existing samples. Where the performance practice consisted of triggering the sampled and sequenced material, creating layers with simple manipulation of the material limited to the sampler's functions such as playback speed, pitch-shifting and in-built or patched in external effects (filters, delays, reverbs etc.) With the portability and power of the laptop, a performance could now be realised using one single portable device.

[The touring rig consist of:] "One laptop computer, a little mixer and a effects unit. But soon I'll be eradicating the mixer and effects. So basically it'll be one computer. It does everything I did before with live samplers and sequencers." (Aphex Twin in: Rule 1997).

Although the use of hardware samplers was perfectly adequate for some artists (like the live sampling work of Jan Bang). Others viewed the laptop as an extension of the recording studio-as-musical-instrument (Irving 2006). An opportunity of using a portable solution that gave the same possibilities for parameter automatisations, complex sequences and manipulation of audio that existed in their computer studio environment that could be exerted live-on-the stage. The software industry was quick to follow up and to capitalise on the popular trends in electronic dance music production and performance. With the emulation of knobs and sliders of studio equipment, the highly GUI centric and studio production oriented plugins or standalone software was being enhanced by adding some "live-performance"

features. These "live-performance" features were typically; dynamic loading of tracks, interactive preset management and sequencers that could be operated on the fly like the popular software Reason released in 2000 by Propellerhead Software. Reason was conceived to be used as a virtual studio or as a collection of virtual instruments for performance (Propellerhead Software AB 2012). Emulating a studio rack of effects, mixers, drum machines, samplers, synthesisers and sequencers with virtual patching cables which could be interactively connected to the devices in an arbitrary manner. Through the sequencing capabilities and the patching combination of the virtual devices the user can switch between different configurations in a performance, effecting the desired changes in the music (Zadel 2006). However they are to be prepared beforehand as presets so that the performer in conjunction with manually manipulating various parameters are triggering the preplanned events or states using the GUI or a external MIDI controller. Artists Robert Henke and Gerhard Behle of the minimal-techno duo Monolake wanted a software tailored for their loop-oriented music and started developing their Ableton Live software, released in 2001 with the catchphrase: "...designed for use in live performance as well as for production " (Ableton AG 2012) emphasizing the previous dichotomy of the studio production and the stage that was now fused in performance. Onstage, this allows Devine to

"...mix and match breaks, intros, or builds for different tracks, and even manipulate how those are played if I select them. I can really do anything with the arrangement of the original track. It is now total remixing and producing on the fly." (Luta 2009).

"The instrument-like character of this technical device, compared to the desktop computer, consist in its personalized mobile use, more reminiscent of conventional instruments" (Grossman 2008: 9). In the fringes of subcultures that grew from the electronic dance music scene, the laptop became central to the very idea of its use as a performance instrument and so closely associated to the music that the moniker Laptopism/Laptop music was being used (Cascone 2000; Kaltenbrunner et.al. 2006). Also theorised and coined as "post-digital" music (Cascone 2000) Laptop music is non-idiomatic in the sense that there is no particular style, where a performer takes on elements and inspiration from a cluster of styles and sources like:

EDM and its derivatives, avant-garde, free-improvisation, sound-art as well from the academic computer music community (Turner 2003; Kaltenbrunner et.al. 2006). As Turner (Turner 2003: 81) puts it: "...post-digital musics stands somewhere between the dance club, the university auditorium and [the] symphony hall".

With the help of the information flow of the internet and a increasing interest in electronic music, audio programming environments like MaxMSP, PD, Supercollider, Csound etc. was spread to music communities outside the academic composer realm and computer music research community (Cascone 2000; Lyon 2002). Countless of laptop artists started using these programming environments for developing their own personalised software instruments and generative tools, custom tailored to their own specific aesthetic ideas.

"I've been doing it for about three months, so it's all quite primitive, but it's looking really interesting. This language you can bring in your own samples and mess around with them. And it's got DSP functions you can't get anywhere else, but you have to program it in. There're no fancy sliders, although they're easy to construct. I've made loads of graphical interfaces for things. The algorithm I just finished... is a percussion thing that lets you swap and change the sounds. It does bass as well, but it's really acidic. You can leave it on for, like, an hour, and it really comes up with some mad shit. I made it learn to gradually change [the music] over time...I just finished a tour, and I used it for one of the tracks. It was pretty interesting watching people dance to my algorithm." (Aphex Twin in: Rule 1997).

Where laptop artists like Aphex Twin and Autechre used their Supercollider and MaxMSP algorithmic creations as a part of their studio-as-a-musical-instrument club performance sets, artists and performers like Kaffe Matthews, Christian Fennesz, Joel Ryan and Lawrence Casserley explored improvisation and performance in the similar vein of the live electronics tradition of the 60s. With the real-time DSP capabilities of their laptops and custom software creations; an input from acoustic sources or other fellow performers could be processed or live-sampled, transformed in real-time, constructing new sounds on the spot and further

manipulate it during the course of a performance. An interesting point to this practice is that the notion of this new instrument is then shared between the computer, laptop musician and the acoustic (input) musician(s).

2.6 Wearing - Off

As the use of laptop and software in electronic music performance and improvisation was becoming more common and less of a novelty, the authenticity of laptop performance and its performance practice was being questioned. The main aspect of a computer is that it's a generalised open ended system i.e. programmable, itself a tabula rasa, full of potentials (Tanaka in Dean 2009: 239). While this gives an enormous amount of creative freedom, blurring the boundaries between studio/composer - stage/ performer and giving birth to new performative modes, it also becomes problematic in terms of the authenticity in performance. With the instrument being in the software, hidden away from the audience, (further not helped by the performer being covered behind a screen with a glowing apple logo), little perceivable coupling between action(s)-to-sound and the cultural implication of the laptop being a general purpose tool equally used for games, writing, browsing, emails as for music; leaves the audience unable to attach value to the experience (Cascone 2003). "Is it playback?", "Is he checking emails while he is on stage?". Many of these issues is being especially true for artist that opts for a "point-click" approach through predefined templates, control and event sequences with very little interaction and engaging activity resembling the issues as in the acousmatic tradition (Cascone 2003; Zadel 2006). Problematic issues is also at place for the laptop performer using studio-centric software with live manipulation of parameters through myriads of GUI knobs and sliders, potentially resulting in cognitive overload. Which in turn might hinder important elements for engaging activity and decisions in-a-now.

2.7 New approaches -from live coding to laptop orchestras

In the past years some quite creative responses and radical approaches to laptop performance have surfaced, such as the live-coding phenomena. Where the execution of the actual coding for sound production and processing is a part of the performance. Using the interface of the coding language itself as a novel performance tool (Collins et.al. 2004). Although graphical based environments like PD/MaxMSP have been used, live-coding is done primarily using text-based real-time audio software such as Supercollider/JITLIB, Impromptu and Chuck. A common practice amongst live-coding performers is also to share their screen by projecting it on stage allowing the audience to both hear and follow the live-coded performance and its development visually. Live coding in performance is not without its issues though; It demands significant amount of programming experience and it might take a long time for a musical idea to be realised, possibly ruling out group improvisation with other instrumental performers unless pre-coded schemes are in place due to the fast paced changes and responses needed in such a setting. Other approaches that goes even further than live-coding, bearing resemblance to the networked approach by LAMC/The Hub has been by the laptop group PowerbooksUnplugged (PbUp). Their name being a play on the highly popular (amongst laptop artist at that time) model by Apple and the live acoustic performance term. By performing "acoustically" using only the laptop in-built speaker they became according to themselves; the first acoustic computer music folk band.

" Many have claimed that 'The laptop is the new folk guitar'; if this is so, then PB_UP is the first acoustic computer music folk band: The laptop is their only instrument. "
(Hoelzl et.al, 2006).

Using self-written Supercollider software and live-coding schemes on a shared network, PbUp also exploited some of the features of the laptop such as in-built speakers, microphone and the wireless network capabilities. By having a wireless network allows PbUp to not only

communicate, share code and send control messages over the network, but also taking the mobility of the laptop to its full use; without being tied to the P.A system or any cables, the performers can move through the performance space, breaking up the audience/performer stage conventions (PbUp). Perhaps one of the most significant and adventurous approaches in laptop performance which have gained interest in recent years has been with the advent of laptop orchestras. The Princeton Laptop Orchestra (PLOrk) initiated by Dan Trueman and Perry Cook in 2005 grew out of a body of research work involving spherical speakers, human-computer interaction design and computer music programming for performance at the Music Department in Princeton University (Trueman et.al. 2006). Taking inspiration from LAMC/HUB, PbUp and other technological oriented groups, PLOrk is based around the idea of creative uses of the laptop and the western classical orchestra. Central to PLOrk is the use of local speakers, with all 15 members using a setup of "orchestral-instrument inspired hemispherical speakers local to each player" (Trueman 2007: 174). Using local speakers gave not only each member a separate voice, but it also made using a studio approach to separating the voices (panning, reverb) unavailable. Meaning that the laptop orchestra shares the same wide distribution of sound sources as is the case with the traditional orchestra (ibid). In addition to the hemispherical speakers each player also have the same setup of audio interface and laptop models. The performance instruments/pieces is programmed in MaxMSP, Chuck or Java where the physical interaction uses the laptop native features: the QWERTY keys, microphone input, trackpad, accelerometer, webcam along with the network capabilities. An example of a direct-action-sound connection and creative use of the laptop's trackpad is in the pieces *Crystalis/Wind-o-lin* by Ge Wang, where the players "bow" the trackpad by varying finger location and the speed to inject "energy" into a synthesis model. Where the different directions of "bowing" places the sound in corresponding audio channels (Fiebrink, Wang, Cook 2007). Other physical interfaces being used in PLOrk is external sensors and devices like pressure-pads, proximity sensor, gamepads etc. (Trueman 2007). More recently they have explored on-the-fly-machine learning for fast and creative mapping strategies using the wekinator software (Fiebrink, Trueman, Cook 2009). A software based on the open source library Weka, for machine learning algorithms where the performer trains the computer to learn and recognise her actions and the desired outcome. Other laptop orchestras

and ensembles have since been founded in many parts of the world, building-on or similar to the PLOrk model such as: The Moscow Laptop Cyber-Orchestra, Stanford Laptop Orchestra, Huddersfield Experimental Laptop Orchestra, Dublin Laptop Orchestra, Oslo Laptop Orchestra, Virginia Tech Linux Laptop Orchestra et.al.

2.8 Post-laptop?

Even with the laptop computer encompassing real-time digital audio processing, synthesis, MIDI and electronic music performance activities into one portable device, the continuation of ubiquitous computing, cost reduction and the miniaturisation of technology have in the recent years led to embedded and mobile devices starting to be powerful enough for exploring computer music performance. As mobile devices in general is a broad term and are essentially small handheld computers, many are naturally designed to be sound-producing devices as in the case of mobile phones and smartphones⁹ (Essl, Rohs 2009: 197) and its bigger "cousin" the tablet computer. What makes them especially interesting and potent for music performance is not only the sound-producing capabilities along with increasingly faster computing specs (CPU, Memory etc.) but the programmability and the interfacing technology they inhibit. Today's mobile devices often features an array of versatile sensors such as: microphones, cameras, accelerometers, gyroscope, magnetometers, GPS and multitouch technology (resistive/capacitive sensors). These sensor capabilities opens up for many interesting ways of interacting with the devices in a musical context as already creatively explored by a number of artists. The popularity of mobile devices for use in performance is rising and while the computing power, audio hardware and software is not as powerful or greatly available as for the current day laptop computer, mobile devices are very capable of doing real-time synthesis and audio and its worth noticing that several hundreds of music

9. The distinction between *mobile phone* and *smartphones* can be very vague although the level of programmability can vary, there is no official definition of what constitutes the difference.

<http://en.wikipedia.org/wiki/Smartphones>

software applications are already available on the two most popular mobile device platforms; Android (OpenHandsetAlliance/Google) and iOS (Apple). With applications ranging from emulations of vintage synthesisers and samplers to creative software controllers and interactive music software. And more recently powerful music programming environments such as Supercollider, RTcmix (iRTcmix), Pure Data (libPd), and Csound (Csound Touch) have also been ported to these platforms, providing an opportunity for musical creations and code to be run on these devices via a more familiar music programming environment.

Physical computing and embedded systems is a burgeon field which have undergone a democratisation process. This has resulted in inexpensive and widely available micro-controllers platforms such as the open-source Arduino and "credit sized" single-board computers like Raspberry Pi with their plethora of extension boards and sensors. Such micro-controllers and single-board computers are easily programmable via high-level software and can be embedded into the material of choice, made to meet specific interactive musical tasks. Low cost and ease-of-use combined with resourceful internet communities makes it easier for the performer/developer to not only create their own software, but also designing the circuits. Thus creating the possibility for highly personal instruments that could exists in both the analog and digital realm, bridging the DIY attitudes of circuit designs and signal chains, as done in the analog live-electronic area of the late 60s with the digital software making process.

2.9 Summarising and discussion

After having taking a closer look at the aesthetic paradigms and historical developments in technology regarding live performance, what is becoming clear is that these paradigms are not only attributable to the technological advancements but also to the rapid expansion of access to specialised knowledge and tools: If we go back to the analog live-electronic area the adventurous people that started using the available technology of the time where very few, needing specialised components, training and understanding of circuit design and electronics in general. With the advent of microcomputers and early personal computers, the shift from analog to digital required costly equipment and highly specialised knowledge of the particular computer architecture. Machine code programming, as little or no high-level sound programming language or software existed outside the academic research community. As the interest in electronic music and synthesisers increased especially in popular music genres, the demand for easier interconnection and usability grew. When the MIDI standard was introduced along with the personal computer revolution there was a rapid expansion of the sales and production of electronic instruments and music software. With MIDI there was now a standardised communication protocol, digital interface and a separation between control input and the sound output. This opened up for interconnections between sound producing hardware and the increasing powerful personal computers. The personal computer revolution also spawned graphical user interface based software and visual programming languages like Max/Msp. Such software removed the need for composers and performers to have highly trained skills in solely text-based computer programming, making it easier to create their own interactive music performance software. The demand now was for real-time audio processing capability in the computer. During the early 90s the cost reductions and advancement in DSP technology, led to real-time audio DSP processors and analog-digital converters being oriented towards the personal computer market. As a result, software synthesisers, effects and digital audio production tools where now becoming integrated in the existing MIDI environments of the personal studio. The personal home recording studio was now digital, with the studio-environment existing virtually in the computer. With real-time

audio capabilities the personal computer was also suitable for live-processing and live-sampling of any source in music improvisation as done by Lawrence Casserley and his work with saxophone player Evan Parker. With the advent of the laptop computer this studio-environment was also now portable, popularised by the explosion of the electronic dance music scene of the 90s. The laptop gave an opportunity of using a portable solution that gave the same possibilities for parameter automatisaton, complex sequences and manipulation of audio that could be exerted live-on-the stage. Throughout the 90s and 2000s the software industry where quick to follow up and to capitalise on the popular trends in electronic dance music production and performance, making studio centric production software tools that could be used "live" in performance. At the same time helped by the information flow of the internet, programming environments normally found in more academic computer music communities was being used by countless of laptop artists for developing their own personalised software instruments tailored to their own specific aesthetic ideas. As the novelty factor of using a laptop in live performance started to fade, the performance practice of using a computer in live performance was being questioned. The laptop performer on stage showed very little perceivable coupling between action(s)-to-sound along with the cultural implication of the laptop being a general purpose tool equally used for games, writing, browsing, emails as for music. Acknowledging these issues, many artists started addressing it quite creatively and this has resulted in interesting aspects of laptop performance practice such as live-coding, laptop ensembles and orchestras and their creative use of the laptop native peripherals. The continuation of miniaturisation of technology, cost reduction and availability have led to mobile devices along with creative use of its built-in sensors and software applications being powerful enough for real-time computer music performance. With the widespread availability of sensors and micro controllers, user-driven electronic interactive prototype platforms are bridging the gap between the analog electronic and digital world, creating the possibility for having personal instruments that exists in both the analog and digital realm. This democratisation of technology, knowledge and information that have been incremental during the last decades where especially the Internet have played a critical role in this process, has given rise to larger activity and wider exploration of aesthetic and performative practice(s). As the computer is a such ubiquitous tool it can be argued as such

that it is the folk instrument of the 21st century, a instrument that anyone could learn to "play" and master in their own specific way. As the computer is a general purpose device, programmable and not bound by any musical or physical constraints as in the case with acoustic instruments there is also no general methodology or any specific How-to-play. This is completely up to the developer/performer of the software instrument, creating a personal instrument where the distinctions between instrument, composition and performance activities is blurred and fused together into one thing. Encompassing the musical activity into a portable and ubiquitous device.

Chapter 3

3 Human computer interaction and digital music instrument design

This chapter present presents basic human computer interaction and digital instrument design. Providing the background and drawing the connection to important issues in the relation to performance and the use of computer.

3.1 On computer interaction, interfaces and the digital music instrument.

Humans can more or less spontaneously deal with several amounts of different processes such as auditive, visual, kinetic, tactile etc. The computer on the other hand is limited to deal with different processes only represented as a formalised “language”, as computer machine code (Lindborg 2006). An interaction involves in the strict sense at least two participants where data or information continuously exchange between the participants¹⁰ (Svanæs 1999). In the context of human-computer interaction, we need an interface, a sort of “translator” device divided into input and output between our real world and the computers digital world in order to communicate and interact. Often we use several different forms of interfaces such as a physical input interfaces like: keyboard, trackpad, joystick etc. For example, a trackpad and the keyboard take the input, a physical action and convert it into sensor data that the

10. There is a number discussions around the concepts of interaction. As its summarizes a variety of disciplines and opinions. I relate very simple to the concept and do not go in-depth into HCI literature or any insight into phenomenological theory (Merleau-Ponty) related to interaction.

computer can understand. The output from these devices is input for the computer. Similarly an interface like the computer screen take input signals that the computer outputs. Then these signals are converted into representations that human users can see for example through the Graphical User Interface (GUI), an abstraction software with graphical and symbolical representations seen on the screen. When different interfaces are combined, it gives us various degrees and types of sensory feedback for the parameters and events we wish to control and manipulate. Such an interaction loop is illustrated by Figure 3.1.

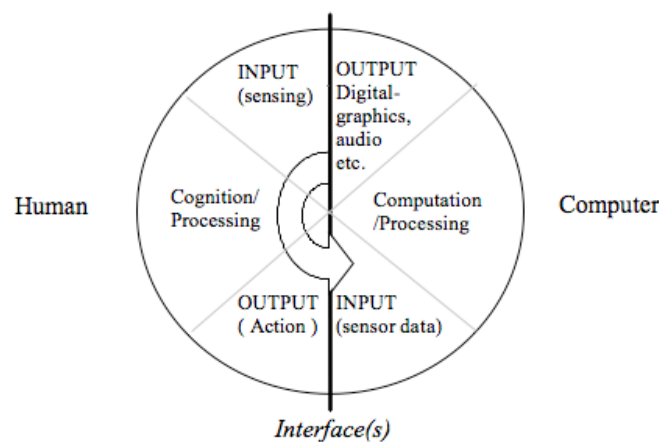


Figure 3.1. A basic figure on an human-computer interaction loop. The output from the computer can be in the range of sensory modalities: visually via a screen, haptic in the case of a force feedback joystick or in the case of music, audio. Or all of them, depending on the number of interfaces and its relationship to the computer.

Since physical action(s) can be converted to discrete or continuous sensor data via input interfaces we can then use this for musical purposes. Actually, any input interface containing a sensor or sensors that can track a performer's actions can be used as a control input for musical applications. When used in such a context the term controller is used. We are then referring specifically to an input interface for manipulating musical parameters. The controller is what constitutes the interface between the performer and the music software on the computer (Jorda 2005).

3.2 The conceptual relationship, constraints and affordances

The computer is not a defined instrument with mechanical and musical affordance and constraints (other than the computational) and unlike acoustic instruments, the relationship action-sound in a digital music instrument is conceptual. There are not necessarily any intrinsically link between the performer action and the generated sound. Meaning, arbitrary connections between action and sound are being made by the designer/developer/musician following his or hers own conceptual and aesthetic goals for sonic outcome. In fact, we may think of the computer as a universal sonic instrument, a subject that we can develop to produce all conceivable (musical) sounds and organisation of this (Ryan in Lindborg 2006). Making this conceptual relationship between action-sound, any linear or nonlinear input connections from a controller are mapped to the sound-engine/computer following a mapping structure such as: one-to-one, one-to-many, many-to-one or many-to-many (Miranda, Wanderley 2006).

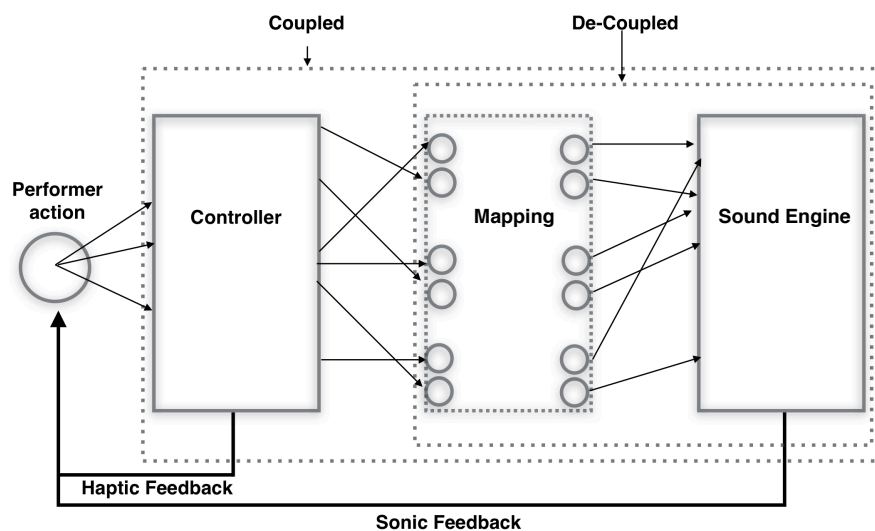


Figure 3.1: A generalised model of a digital music instrument. Showing the action-(mapping)-sound relationship in coupled (unified) and de-coupled approach.

As such, a conceptual relationship of action-sound makes it possible to create highly personalised instruments, and even possible to dynamically change the action-sound relationship on-the-fly through re-mapping the connections. However, as pointed out in the previous chapter (2.6) there might be certain problematic issues that arise when using the computer as an instrument in musical performance. If opening up for control of every parametric possibility it is easy to get cognitively "lost" in the myriads of buttons, knobs, sliders or any other parametric control mechanisms, resulting in an un-engaging interaction. Or at the other end; too constrained, the interaction level would be low or un-engaging. For example, mouse-clicking through predefined templates, control and event sequences etc. Therefore, important questions when designing the conceptual relationship action-sound are: What kinds of action-sound does an interface/controller allow, suggest, or invite to? And equally; what does it not? As all tools or instruments has designed usage and transformed usage it can be described in terms of affordances and constraints (Magnusson 2006). As the term *affordance* is a much debatable subject and surrounded in ambiguity it is completely out-of-scope to be discussed in any depth in this thesis, I will however highlight the two most distinct views:

The term *affordance* was originally coined by psychologist J.J.Gibson in his book *The Ecological Approach to Visual Perception* (Gibson 1979) where he defined affordance as what the environment offers the individual (ibid :127). Meaning that affordance exists independently of the individual's ability to perceive them and do not depend on previous experiences and culture (ibid). The term was later adapted and consolidated into the field of Human Computer Interaction (HCI) through the work of Donald Norman, where Norman in his book *The Psychology of Everyday Things* (Norman 1988) interpreted Gibson's affordance to include perceived (action) affordance. I.e suggestions or clues as to how to use the properties or take action upon an object, and in conflict to Gibson definition; based on experiences, knowledge and culture. A much used analogy of these two views is the chair example. According to Gibson a chair's affordance is to be "sat on" but can be "thrown", "stood on" etc. While Norman's view is based on the perception of the object, the chair is to

be "sat on" according to the individual past experience, knowledge, culture and so on. Norman argues that for example a button in GUI or anything that appears on a computer screen is a perceived affordance (pushing the button), but the actual affordance properties lies in the computers physical components actual clicking the "button" through the use of the trackpad, mouse or keyboard. What is interesting in this case when designing digital music instrument is that Normans definition of affordance is related to the idea of constraints (Magnusson 2006). Whereas Gibson's view would be perhaps more from an ecological perspective. Even though there might be different types of constraints (Norman 1999) a generic definition is that constraints limit the way an object can be used or viewed in certain capacities, in opposition to its affordance. For example, a normal guitar has six strings in which it has certain constraint or limitations when it comes to chordal voicing possibilities, pitch-range, number of sustained notes etc. However, in music and creative practices constraints could be viewed as a way of augmenting affordance. The guitarist might after spending time with the instrument explore its affordance but then find ways of thinking-out-of-the-box via specific practice, extended techniques or via a completely transformed use of the guitar. Thus creating affordance out of the constraints. Therefore, constraints can possible create a search space in which types of creativity can take place (Boden in Magnusson 2010: 64). In digital music instruments as Figure 3.1 indicates, constraints and affordance can exist at several stages: At the controller/interface stage (its physical/technical properties) and also at the mapping - sound engine stage (artificially constructed). In the design of digital music instrument there are different views on what constitutes a digital music instrument and as such where the affordance/constraints is truly located. Thor Magnusson argues in his article *Designing Constraints: Composing and Performing with Digital Music Systems* (Magnusson 2010: 65) that given the de-coupled nature of a digital music instrument, the constraints are defined and located at the mapping - sound engine stage as this is the core for where the "real" body of the instrument is. For Magnusson the controller is merely a mechanism providing affordance for the physical action (ibid). Others, such as Jorda (Jorda 2005), advocates a more holistic or unified view on the digital music instrument as a coupled system where the controller, mapping and sound engine is the instrument and as such constraints is also manifested at the controller.

Another interesting point to this discussion is with regards to the term *disfluency* a newer term stemming from cognitive sciences which defines it as "the experience of processing difficulty" (Thompson, Ince 2013 in Bin, et.al. 2018). The term has recently been adapted and used in the field of digital instrument design (Bin, Kinns, and McPherson 2018). Disfluency and constraints are both referring to a limitation but where for example an interface will remain constrained, disfluency will diminish when the element becomes known and the performer adapts to it and it becomes second nature (ibid : 2.2). The disfluency is then a state with levels of unknowns. Constraints can be potentially important in order to spawn creative artistic output. Designing an digital music instrument specifically oriented towards the context of live music improvisation, constraints could be a key factor using it in a disfluent context such as improvisation. Where the instrument should provide a level of encouragement and exploration to out-of-the-box thinking and transformed use.

Chapter 4

This chapter provides description of the development, implementation and general documentation of the software. There will be a short introduction to the aesthetic and or technological objectives of each instrument, followed by a description of its use and a brief outline of the artistic outcome.

4 Laptop Instruments

By taking the concept of constraint and affordance in a unified instrument approach by using the laptop as the physical controller and sound-engine, I wanted to explore this limitation in the context of live music improvisation. As this context was a key premise for the exploration, it meant that they had to be designed to facilitate an engaging activity in the interactive relationship and alleviate any technical distractions from the possibility of an intensified "now". An intensified "now" or a subjective "now", to use a term by Edmund Husserl without going into his theories, is the very fundamental concept of improvisation - it is a focus of the activity itself, a place where actions, both past and future appear in this very now moment. (Gustavsen 1998).

Rather than combining or using all the interfaces available in a laptop all at once for creating an instrument I chose to focus on the individual native built-in interfaces, their properties and physical constraints. This gave me a better chance to evaluate each interface/controller, and also was a way to further constrain the instrument and force myself to think outside-the-box. I will stress that I emphasise the use of the term "laptop instruments" rather than digital music instrument in this and the following chapter in order to be precise and specific about this

unified approach. All the instruments described run on Mac OSX only (10.6 and up) for the MacBook laptop models range and are developed in Max, a visual programming language for music and multimedia by software company Cycling74 (www.cycling74.com). Max was chosen as the development platform due to its suitability as a modular prototyping platform, as well as its functionality and ease of use. The instruments are made primarily as maxprojects which are contained collections of the patches, code, and other files needed in order to run the Laptop instruments. In some cases the instruments will also run as so-called "Max devices" in Ableton Live, these are also included in max-project files where applicable. Although the general functions are being described in this section, opening, running, and experimenting as well as reading contextualised comments in the patches might further clarify the detailed functioning. Understanding the descriptions, patches and code requires some knowledge of the Max language. Versions 6 and 7 of Max for the Mac OSX Operating system has been used during the development. The latest version of Max at this time of writing, Max version 7.3.5 should be run in 32-bit mode for best compatibility as some of the instruments uses third party extensions that for the time being only exist in 32-bit.






4.1 The Keyboard variations

One of the Laptop's major control interfaces which exists regardless of model and make is the built-in keyboard. The laptop's built-in keyboard is simply an array of switches reporting a binary state per key. I.e. pressed or released. The very minimal physical feedback to the user is the mechanical response of the keys pre-travel and release travel (how far to be pressed and when to be released) This is also indicated by sound, the click-sound when pressing and releasing a key and for some keys, additional LED lights. Asking the question: "What kinds of action-sound does an interface/controller allow, suggest, or invite to? " Being an array of switches the only physical actions possible is by pressing/releasing one or several keys. Either at the time and or in specific pattern/sequence(s). At the same time the keyboard invites to be typed on, inputting text, numbers and characters used in the same way that I am writing this thesis. By using its text input affordance and impulsive-iterative action type as a

point-of-departure, I have developed several laptop instruments using the keyboard as the controller. Exploring variations of action-sound relationship and the artistic use in improvised music contexts. Three of these laptop instruments are described in-detail in this section. The following sections describes their general functioning, process and comments surrounding their use in performance and qualities.

4.1.1 αβ

αβ (AlfaBeta) was originally conceived as a laptop instrument specifically tied to an open form composition of the same title. The composition is a play on a typical laptop music performance; to the audience the performer might be checking e-mails or participating in chat. So the idea was simply to make a musical performance out of a chat where the amplified acoustic click sound of typing on the keys was the sole source of sound. In the αβ instrument the acoustic sound of a key on the laptop's built-in keyboard is instantly sampled when pressing a key down, recorded through the internal microphone, contact mic or by other means and played back instantly. The capture of the key data for controlling the recording/playback and all other functions in αβ is done by using a built-in KEY and KEYUP object in Max. These objects read, respectively the pressed and released key as ASCII values. The ASCII values are then being converted and sent to a Javascript where a lookup table/array checks which values are coming in, and replaces the ASCII values according to the index in the array to represent the numbers 1-46 and their state. These are then passed on as triggering messages for recording, sample playback and for other logic connections made in the patch. In order to contain the click sound of pressing down and releasing the key, the allocated sample memory in the recording buffer is constrained by design to 1 second. Working as a sort of micro live-sampler but with no looping possibility, sound is only emitted via a *one action - one sound*. If no typing action is made: no sound. The number of action(s) is then somewhat proportional to the number of sonic outcomes. In αβ 46 keys of the keyboard have what I call "active functions" (29 letter keys, 10 number keys and 7 character keys). The rest of the keys do not have any functions other than outside the Laptop instrument, an exception

to this is the  which is only used in conjunction with the  key. The 46 keys with active functions are grouped into four Varispeed Playback modes, given in speed ratios / pitch intervals: Loco (1:1 / 0), 8vb (1:2 / +12), 8va (2:1 / +12) and Reversed (1:-1 / 0). Each of the 46 keys triggers a different filter effect using various second order IIR lowpass, highpass, resonance, band pass, and notch filters that the sampled sound is run through. Furthermore a subgroup of keys activates an additional simple reverb with nine different presets of parameter settings. Finally, the + key combination produces  (exclamation mark) and triggers a filter-sweep based on the filter of the previous key. Perhaps the most novel feature is the addition of "word typing based audio effects" where up to six defined words or sentences (decided by the user), when being typed triggers six different DSP effects that are applied at the end of the signal chain, these are: Stuttering effect, Granulation, Frequency shift, Feedback Delay, Wavetable based FX, and Ring modulation. The various parameters of the six effects are dependent on the speed of typing the words/sentences. Typing faster, the effect/phrase happens quicker and vice versa. This is implemented in the following manner: The typing speed for each character matching the target word is tested against the number of characters that constitutes the defined word. The accumulated speed is then used to change the time of the parameter in question and the effect passed/bypassed. Another typing-speed mapping is also used on some parameters of pitch/frequency changing types where the typing speed is scaled against a measurement based on Characters per minute (CPM) x Length of word (L(n)). The total typing time is scaled against a Minimum of 23 CPM (Characters per minute) = 526 ms. x L(n) and of the Maximum of 1060 CPM = 56 ms. x L(n). The result is then converted to the required parameter range of the effect. The very execution of a complex word or the length of a sentence is then being linked to the resulting sound as an aggregate function. $\alpha\beta$ in its performance was originally set up so that several laptop performers, each running $\alpha\beta$ instrument on their laptops, could connect to a network, join in on a chat channel which were integrated into this instrument and chat/type the actual performance.

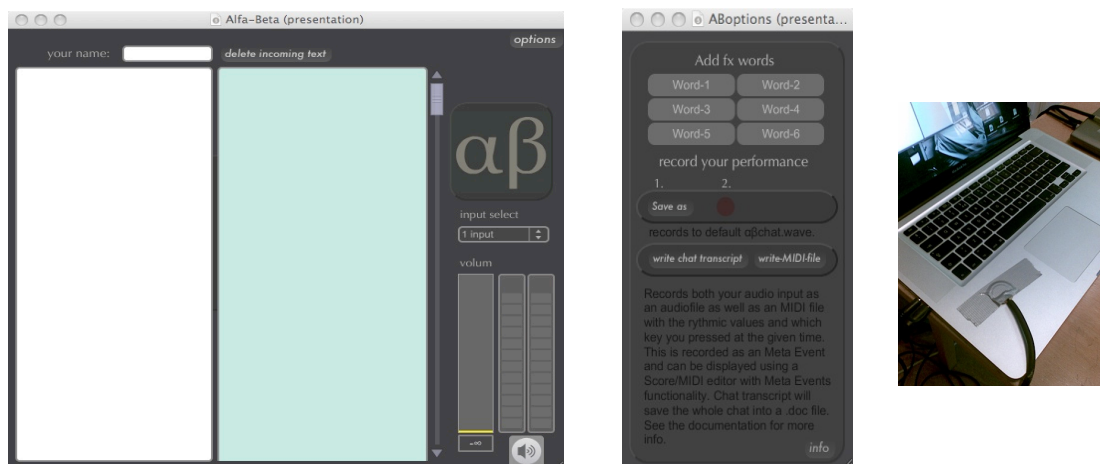


Figure 4.1.1. $\alpha\beta$ Main chat interface along with the word/sentence definer. To the the right, a contact mic placement on the laptop chassis for capturing the keyboard sound.

4.1.2 $\alpha\beta_Redux$

For an adaption of $\alpha\beta$ to a live instrument the use of the keyboard was something I thought had some very interesting features. The simple, direct and very quick way of live-sampling, and the connection to a typing paradigm, was something I really enjoyed working with. I modified the $\alpha\beta$ slightly with regards to using it in my improvisation performance practice. After making some improvements to the DSP and dropping the chat functionality, as this was redundant, it developed into a new variation of the instrument which I entitled $\alpha\beta_Redux$. The GUI interface is kept to a very minimum with only a menu for choosing I/O of the audio interface, setting the sound level output, and the option for six defined FX words. As with the original instrument it advocates a fast way of live sampling, but instead of focusing on the QWERTY keyboard clicking sound material I used it for capturing other acoustic/electric performers via external microphones or contact microphones, in a similar fashion to Joel Ryan and Lawrence Casserley's work with the Evan Parker ensemble. Based on some early experiments I expanded the four varispeed groups to seven in order to have slightly larger distribution in speed/pitch playback. Given in speed ratios / pitch intervals: Loco (1:1 / 0),





8vb (1:2 / -12), 8va (2:1 / +12), Reverse (1:-1 / 0), 16va (3:1 / +24), 8va-Reverse (2:-1 / +12) and 16vb (1:3 / -24). I also changed some of the "word typing based audio effects" to be more aligned with what I personally liked for audio processing different acoustic material, especially since the frequency content would be far more varied than click sounds. They do however, keep the same typing speed dependencies and other functionalities as described in the above section. The word-based effects in $\alpha\beta_Redux$ are: Spectral Blurring, PitchDelay, Granulation, Comb filter, FeedbackDelay and a FilterDelay. The same direct, and quick way of live sampling whilst keeping to a *one-action-one sound* paradigm proved itself to be interesting and fruitful in this case as well. Especially having the sample recording memory constrained to 1 second provided for an engaging activity, constantly finding ways (musically) of fighting these limitations through the use of multiple keystrokes, manually creating rhythmic phrases by iterating keystrokes fast enough before the recording sample memory was being cleared (1 second after a key release). However, I was found it more useful to sample percussive type sounds from performers, as longer, more sustained sounds, would tend to be slightly disrupted or difficult to use or maintain at times.



Figure 4.1.2: $\alpha\beta_Redux$. The 46 keys with active functions.

4.1.3 $\alpha\beta$ _polypoly

Through spending time performing with $\alpha\beta_Redux$ in various artistic settings it became clear to me that it was not always practical, nor was there sufficient time for setting up external microphone lines from my fellow performers' instruments, as the set-up required a strategic positioning of the microphones (close-micing), and a careful adjusting of input gains and levels to get the desired result. Moreover, I wanted to experiment on how I could extend the same idea of using the keyboard as previously, but without live-sampling external source material, choosing rather to utilise pre-sampled material, i.e. a sample-bank. Hence $\alpha\beta_polypoly$ was conceived where a sample is triggered when a key on the laptop keyboard is pressed. The idea of triggering a sound or a sample with the QWERTY keyboard is definitely not new, and it's a relatively old way of auditioning a sample or trigger sound on a computer. Take for example Ableton's Live software where one has the option to use the key mapping mode to map a certain key on your computer keyboard to trigger a sample clip or midi clip. However, the tighter integration of the keyboard as used in my former $\alpha\beta$ instruments along with the word typing based audio effects, was something that I would try to maintain in this variant as well. And rather than making elaborate and complicated ways of achieving similar results in Ableton Live, I preferred to contain the sample bank in the instrument. In the two previous versions of my instrument, one of the designed constraints was the allocated sample recording memory of 1 second. As $\alpha\beta_polypoly$ does not use any audio input I needed to think differently in order to have a similar time based constraint. From my experience of the two previous, the $\alpha\beta$ and the $\alpha\beta_Redux$ I felt that keyboard as controller lent itself to shorter and more percussive sounds. I therefore ended up with designing a sample bank consisting of 138 short percussive sound materials generated from various sources of acoustic and electronic sounds, with a duration of 40 - 1000 ms. at 44.1 kHz Sample Rate. This new variation of the keyboard-based laptop instrument is basically a multitimbral polyphonic sample playback instrument where each key triggers a percussive sound from the sample bank. The polyphony is constrained to 10 voices, for a possible 10 finger keystrokes. The

core of the multitimbral polyphonic sample playback engine is partly inspired from Peter Batchelor's *Clatter* range of Max patches (<http://www.peterb.dmu.ac.uk/maxClatter.html>). As used in my previous incarnations, the same filtering function per keystroke (they can be bypassed as an option) and word-based audio effects from *αβ_Redux* is maintained. In *αβ_polypoly* the 138 individual samples are distributed to three layers of 46 active keys at the time. Layer 1(default): samples 1- 46, Layer 2: samples 46 - 92 and Layer 3: samples 92 - 138. Shifting between layers is achieved by using the  keys for shifting down/up a layer. The distributed samples amongst the 46 keys are following the pattern as depicted in Figure 3c. The Vari-speed functioning of speed/pitch in *αβ_polypoly* differs from the previous *αβ* versions as it is mapped following the same pattern of sample distribution. Where the  key provides the lowest (8vb) transposition and the  key the highest (3 octaves + one 6th. up). In order to transpose the distribution globally, the  keys transpose up/down in semitone increments. Rather than making it possible to switch or load new sample banks dynamically I chose to make the pre-made sample bank embedded, by design, into the instrument. I made this limitation in order to fully explore this very specific "palette of sounds". The 138 samples provide a good number of variations, and if one includes all the possible transformations of performing with *αβ_polypoly*, it is vast. However tempting it would be to switch to a new sample bank or make it dynamically changeable, I have, through artistic implementation, found number of ways of using it without really losing the "freshness", which often can be the case when the novelty factor starts to wear off.

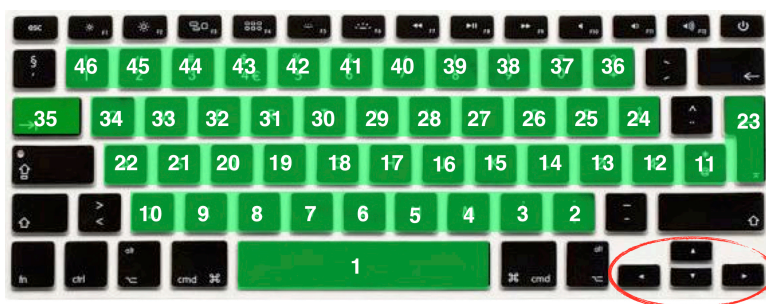


Figure 4.1.3. Sample distribution in *αβ_PolyPoly* to keys, this also corresponds with the transposition 1 - 46 in MIDI semitone increments with a MIDI pitch base of 47. In speed ratios and pitch intervals, ranging from: 1:2 / -12 to (6/7:10):1 / +43). The arrow keys, where left/right shift between the three layers, have been circled in red. Up/down arrow keys transpose correspondently in semitone increments.

4.2 Summary of the Keyboard variations

The three keyboard-based laptop instruments as described uses the keyboard as controller to record, trigger playback and control various audio processes by typing the keys. Providing a one-action-one sound based on the keyboard constraint. Another common factor between these three are a novel feature entitled "word-based audio effects" which forms an aggregate functioning to the sound. Different selected audio effects are linked to user defined words or sentences. In turn the accumulated typing speed of these user defined words or sentences determines the rate of change for specific parameters of the audio effects. Similar types of *speed-to-parameter* changes are also achieved with a measurement of speed calculated by using Characters per minute units x Length of word. Two of the Laptop instruments $\alpha\beta$ and $\alpha\beta_Redux$ are variants of the same idea in that they use live-sampling and are constrained by a sample record memory of 1 second. The third $\alpha\beta_Polypoly$ differs from the two previous as it uses a pre-made sample bank consisting of 138 short percussive sounds. The constraint by design lies in the embedded sample bank and the short length (< 1000 ms) of the individual samples. All three keyboard-based laptop instruments have been used artistically in various contexts. Through artistic exploration of the instruments their fast, effective way of live-sampling or triggering samples albeit limited in duration lent itself towards a more percussive use, as reflected in the physical action.

4.3 The Multi-Touch instruments

The trackpad (or touchpad) is another of the Laptop's major control interfaces regardless of model and make. The trackpad is an input device which features a tactile sensor and a tracking surface. A generic trackpad tracks the two-dimensional position of a user's fingers on the trackpad, as a pointing device relative to the position on the laptop / computer screen. (<https://en.wikipedia.org/wiki/Touchpad>) However, some trackpads such as the Multi-Touch trackpad found on Apple's MacBook range of laptops offers functioning beyond this. Which in addition to tracking multiple fingers (Multi-Touch) include: pressure, size of point, rotation, angle, and velocity. It should be noted that Apple uses the term Multi-Touch as expressly linked to their specific use of predefined gestures-to-subroutines in their OS. (<https://developer.apple.com/design/human-interface-guidelines/macos/user-interaction/mouse-and-trackpad/>) However a more generic definition is: "...multi-touch is technology that enables a surface (a trackpad or touchscreen) to recognize the presence of more than one point of contact with the surface" (<https://en.wikipedia.org/wiki/Multi-touch>). Since the trackpad tracks a physical motion it is, in essence, a motion capture device and might suggest a number of possible actions (Jensenius 2007). As such the trackpad makes it interesting for exploring its potential as a musical controller. Where the continuous data of the motion tracking can be used as several sound-producing actions. In the case of the Multi-Touch trackpad, from multiple fingers and other types of data values harnessed from this interface. I developed a number of tools and instruments based on the built-in Multi-Touch trackpad, three of which will now be described more in detail.

4.3.1 at-multitouch

Since a number of actions is possible using the Multi Touch trackpad, implying continuous actions - sound but also impulsive and iterative types, I developed a controller tool, *at-multitouch* in order to explore the various sound-producing actions. The tool is the core functioning embedded in the other developed instruments described later in this section, but it can be run as a standalone program for making connections to sound-engines outside Max. The tool is quite extensive so a more technical description is in place. The *at-multitouch* access a number of raw data streams from the Multi-Touch trackpad. This is done using the third-party Max external Fingerpinger (<http://www.anyma.ch/2009/research/multitouch-external-for-maxmsp/>). This in turn is based on the hack code from <http://www.steike.com/code/multitouch/>. The raw data stream is read at a rate of 125 Hz for each finger from the Multi-Touch trackpad via the fingerpinger external, it is then scaled and normalised with an optional amount of smoothing of data in *at-multitouch*. The data for each finger comprises of:

- *Finger*: ID of each touch of finger (0 - 3) int. Technically it can track up to 11(!) Here its constrained to 4 fingers due practical implementation.
- *Angle*: Angle (Rotation) of finger(tip). (-1, 1)
- *Major_Axis*: Axes of the finger ellipsoid.
- *Minor_Axis*: Axes of the finger ellipsoid.
- *X_position*: The horizontal position of finger on the tracking surface. (0, 1)
- *Y_Position*: The vertical position of finger on the tracking surface. (0, 1)
- *Velocity_X*: The velocity of the horizontal position per finger. (-1, 1)
- *Velocity_Y*: The velocity of the vertical position per finger. (-1, 1)
- *State*: Tracked finger on or off the surface. (0 - 1) int
- *Size*: Amount of touch/finger. (0, 1)

Derived from this data I also added in the following:

- *Magnitude*: The Magnitude of finger 2D velocity. $[\sqrt{Xv^2 + Yv^2}]$. Norm. (-1, 1)
- *Kinetic_Energy*: The Kinetic Energy $[0.5 \times m \times (V_{xy}^2)]$ where $m = Size$. Norm. (0, 1)
- *Acceleration_X*: Acceleration of the X position (second derivative). (-1, 1)
- *Acceleration_Y*: Acceleration of the Y position (second derivative). (-1, 1)
- *Azimuth*: Of the finger placement on the trackpad surface. Counter-clockwise (1, -1).
- *Centre_to_Edges*: A finger closeness on the surface from centre to far-most edge (0, 1)

The parameter values are sent out as Open Sound Control Messages (OSC) following the namespace nomenclature */at-multitouch/fingerID no. /the-parameter-name* at port number 8100 at the localhost ip address ,127.0.0.1. To add flexibility, a MIDI mode where the conversion of data to MIDI CC control messages is also implemented. MIDI Notes is available but only for the X_position. The pitch range of the MIDI notes is constrained to a range of ten semitones (major sixth) which reflects the relation to a finger limited physical placement on the surface forming a "pitch grid". See Figure 4.3.1. The pitch range can be transposed however by using the transpose function in GUI. In order to have a continuous, microtonal pitch functionality of MIDI notes, a 14 bit pitch-bend is used. Furthermore, an option to round off the pitch in a "two-way rounding" is present:

- 1: Rounding off a starting tone to exact equal tempered tuned note then released for gliss/pitch bend and rounded off/snapping to nearest smoothed equal tempered tuning.
- 2: Rounding a starting tone to an exact equal tempered tuned note, then released for gliss/pitch bend freely with no rounding.

In the MIDI mode each finger corresponds to a MIDI channel, starting at channel 1.

It should be noted, with regards to the MIDI mode, that at the time of the development the MPE (MIDI Polyphonic Expression) specification did not exist. The MPE specification would certainly alleviate many of the workarounds I have implemented in *at-multitouch* as it: "...enables multidimensional controllers to multiple parameters of every note".

(www.midi.org). However, MPE will be implemented in future work.

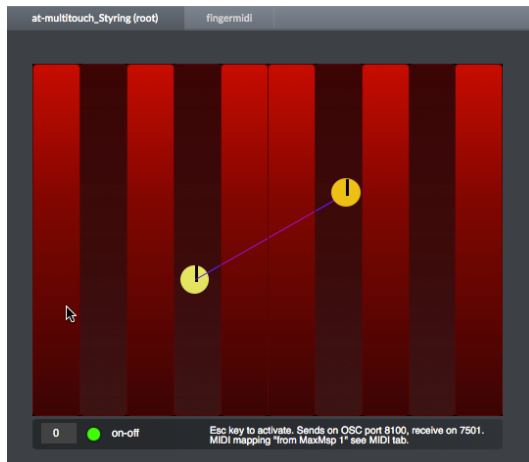


Figure 4.3.1. The *at-multitouch* "performance space".

Additionally, a small Max abstraction *at-mapping* which has a graphical breakpoint function for custom scaling or remapping the parameters can be used as a drop-in placement before mapping the parameters to the sound engine. The abstraction reads either MIDI or OSC and passes the new re-mapped data on to the engine. This can be useful for creating non-linear interpolation curves.

4.3.2 *at-crossConv+Sampler* = Continuum?

One of my early inspirations for using a Multi-Touch trackpad, especially with regards to explore its use as a continuous (pitch) control, is the Haken Continuum Fingerboard. A Multitouch and pressure sensitive digital music instrument developed by Lippold Haken (<https://www.hakenaudio.com>). A DMI associated with its expressive performances with very precise control for amplitude (pressure), continuous pitch (horizontal), and timbre morphing (vertically). Using the Multi-Touch trackpad as a "poor man's Haken Continuum" I wanted to explore some similar types of expression of continuous pitch and timbre morphing. The following description of such an exploration consists of the *at-multitouch*, a standard

sampler, and the *at-crossConv*. The *at-crossConv* is a spectral cross-synthesis effect with a built-in simple monophonic sampler with time stretching and transposition capabilities. The result of a cross synthesis technique creates a timbre "morph" between two different audio signal spectral content, where 0 is the clean source signal and 1 is the clean target signal, and anything in between would be a spectral cross synthesis of the two. The *at-crossConv* is using a fairly standard method for cross synthesis where the built-in sampler, signal B, is convolved (multiplied) with adjustable portions of the input from an external sound source, signal A, or vice versa. The following method as implemented in *at-crossConv*: A real-time FFT analysis is done on each signal, A and B where the amplitude and phase is separated for each signal and then crossfade mixed/ interpolated into each other respective amplitude and phase content of the spectrum before the inverse FFT. The amplitude and the phase of both, source (A) and built-in sampler (B) can be interpolated independently or be linked together. The interpolation of amplitude and phase can be done either: linearly, logarithmically, exponentially or by a graphical breakpoint function. The FFT size of the analysis in *at-crossConv* is set before hand, "hardwired" in the system, and cannot be changed dynamically. In order to play with different sizes, so to be able to accommodate for different frequencies and time resolution content, *at-crossConv* exists in versions for FFT sizes of: 512, 1024, 2048, 4096, and 8192.



Figure 4.3.2a The *at-crossConv* with a basic sampler as source A.

By using another straight forward sampler as the external source A, a simple yet effective way of achieving a continuous pitch and timbre morphing instrument was by mapping:

- *X_position*: → pitch/transpose in sampler A (source) | transpose sampler B (at-crossConv sampler) In case where an external sampler is to be used for A, the pitch transposition can be done using the 14 bit MIDI pitch bend implementation by activating the MIDI note mode of the *at-multitouch*.
- *Y_position*: → interpolation of amplitude and phase between the two signals, I.e the morphing between spectrums. Set at a Log. type.
- *Size*: → amplitude of both samplers (A and B) | → Time-stretch in the sampler (B).
- *State*: → play/stop or note-on/off.

The *Size* parameter is a compromise for the lack of pressure sensitivity or a Z position. And this comes at the expense of a side-effect. Since the *Size* parameter tracks the amount or broadness of the finger used on the surface, it will also slightly affect the Y axis. However, this issue can be alleviated to some extent by scaling, for example, the connection from the *Size* parameter exponentially. Or perhaps a more pragmatic approach is to just consider this side-effect as a part of the peculiarity of the timbral and musical expression in the overall action-sound relationship. Using this as a continuous pitch controller is limited compared to a large-scale Haken Continuum, due to the small surface of the trackpad. The practical number of using several fingers for polyphonic continuous pitch, based on experience, is limited to two. The pitch range of 10 semitones is also limited but can be alleviated to some extent by transpose messages. Despite these constraints the combination of the continuous control, cross-synthesis and a sampler is still quite powerful and surprisingly expressive. Even by just using one finger on the Multi-Touch trackpad one is able to attain quite a high level of fluency with the material. The key to this, in my experience, is by using non-linear mapping of the trackpad parameters to the sound engine using the at-mapping abstraction or by other means between the two stages. Especially when using concrete sounds where a "morphing" between such sounds and their transpositions and pitch fluctuations are making

for a very organic way of improvising with alluring pitch content.



Figure 4.3.2.b To the left: A custom sticker on the trackpad surface visually mimics the playing surface of the Haken Continuum (right). The sticker reflects the pitch grid as depicted in Figure 3.2.2.

4.3.3 at-granular

In contrast to the pitch paradigm explored using the Continuum inspired approach I wanted to investigate how one could use finger motion to explore temporal aspects of sound. A powerful and much used audio processing and synthesis technique is granular synthesis - in this case a sample-based granular synthesis. A granular synthesis segments a sound into grains where each grain of sound is a very brief micro-acoustic event. (Roads 2001). A granular synthesis technique combines thousands of these grains over time and by adjusting selected parameters to its algorithm, it reassembles the sound into a new time order, creating new textures in a fluid-like motion. The *at-granular* is, as the name implies, a granulation-based instrument. It uses four sample-based asynchronous granular synthesis modules, made using IRCAM's mubu toolbox for Max (<http://forumnet.ircam.fr/product/mubu-en/>). Each granular module corresponds to a finger where its parameters are controlled by the action produced by the finger motion. The movement of the fingers on the trackpad surface is used

to control grain parameters, as opposed to a more traditional use where these parameters are controlled often using random or periodic functions, making way for performative actions and an engaging activity. The parameters controlled in each granular module:

- *Frequency*: The frequency of grains, in Hz.
- *Position*: Position in time of the selected sound file. The range is scaled from the duration of the selected sound file in the module.
- *Position Var*: Variation in position.
- *Transition time*: This is the time in ms. for the transition to a new position.
- *Duration*: The duration of each grain in ms.
- *Output Delay*: The delay time in ms. between each granulated output channel. Creates further decorrelation and spatial enhancement.
- *Play*: Start or stop the engine.

The mapping:

- *Position_X* → *Position* (0 - duration of sample in ms.)
- *Position_Y* → *Frequency* (3 - 40) Hz
- *Velocity* → *Position Variation* (80 - 1000) ms.
- *Size*: → *Duration* (10 - 1000) ms. | → *Output Delays* (0 - 600) ms.
- *Kinetic_Energy* → *Transition time*: (0 - 1000) ms.
- *State* → *Play* (0 - 1)

In *at-granular* a selectable folder with sampled sound is loaded at the start. Any of the sound files in the folder can be selected into each of the four granular modules with corresponding sample buffer. But it is also possible to record / live-sample external source material in real-time by selecting which of the four buffers the sound will be placed and recorded into. This makes for some interesting combinations of granulated pre-sampled material and granulating live sampled sound. Again, in early experiments I opted for a simple approach to mapping the data from the trackpad to *at-granulator*. Although I have since made several variants and

changes to the mapping described here, it is nevertheless a starting point for exploring finger motion and microsound.

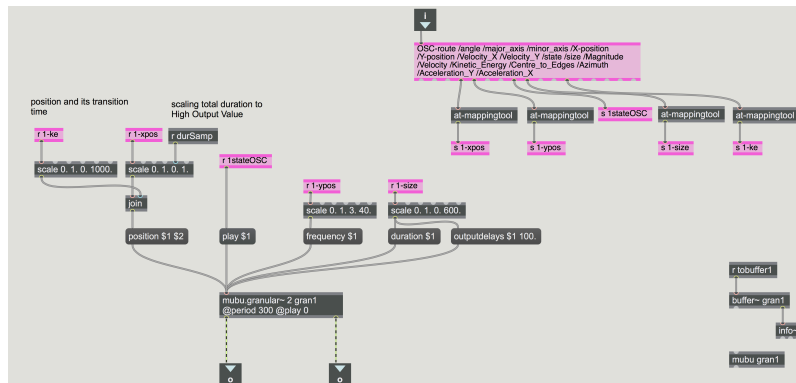


Figure 4.3.3. *at-granular*: A granular module, its connections and mapping.

The powerful sound shaping provided by granular synthesis as used in *at-granular* makes for a very flexible instrument, a vast range of possible textures and timbres is literally at one's fingertips. However, it is not necessarily a straight forward action-sound relationship. Firstly, it is very dependent on the source sound used, secondly an impulse action like a tap on the trackpad could produce impulse or iterative type sounds, but a continuous finger motion could, in addition to producing a sustained sound, also produce impulsive or iterative sounds. So, the action - sound relationship is therefore ambiguous or unclear. But the very idea of motion controlling temporal aspects of sound is what I found the most interesting using this instrument; the fluid way of shaping the sound with the finger motion. Dependant on the sound file used it is an engaging instrument to use especially when calling for idiomatic electroacoustic "gestural" dynamic type-of-sounds. It also provides a way of coaxing new sounds, timbres, and textures from even within an all-to-familiar sound. I rarely switch any the sound files I normally use, because there is always something new to discover; the nuances of a finger motion are strongly reflected in shaping the nuances of the sound.

4.3.4 Other variants

In addition to the aforementioned instruments, I developed a number of different variants of these instruments, or used them in combinations such as *at-crossConvGRAN* which combines the cross synthesis engine from *at-crossConv* and a granular module from *at-granular*. The functionality of both have already been described, but the reader is advised to open and experiment this hybrid version. Another example can be seen via the small and simple experiment *at-noise*, where multiple fingers are mapped in various ways to control filtered noise. *at-noise* uses a white noise oscillator which runs through a State Variable Filter, using the *gen~* object in Max. Setting the various filter modes (lowpass, bandpass, highpass, and resonant) manually. The mapping is as follows:

- *Size* finger 1 → Q factor of the filter.
- *Magnitude* finger 1 → smoothing of data at 150 ms. → amplitude
- *Y_Position* finger 2 → frequency of filter.

The idea is firstly to explore the magnitude of motion, injecting more energetic motion with using more "pressure" being the *Size* parameter increasing its amplitude and resonant qualities. Secondly an approach to using another finger to control another quality of sound. The result is quite clear and a direct action-sound relation with the magnitude of motion being the contributing factor.

4.4 Summary

Of the three creations described, the *at-multitouch* can be seen as being the bridge between the sound producing instruments. It functions as a harness the raw data from finger motion done the Multi-Touch trackpad, derive other types of features from the raw data such as kinetic energy, magnitude, acceleration, azimuth (on surface), centre-to edge (on surface), and total velocity. It normalises the data and outputs the new data formatted as OSC messages or MIDI CC and MIDI notes. The *at-multitouch* is embedded in the instruments or can also be run separately to be used with other sound engines/ softwares using OSC and or MIDI. The *at-multitouch* also provides a visual indication of playing surface. Moreover, a small Max abstraction at-mapping tool was made and can be inserted between the data output and input to sound controlling parameters for non-linear interpolation curves. The two sound producing laptop instruments using Multi-Touch trackpad differ in approaches and explore two ideas: finger motion-controlling pitch and finger motion controlling time. Where *at-crossConv+Sampler* is taking inspiration from the Haken Conitnuum Fingerboard and is a continuous pitch controller with 'sound morphing' capabilities using a cross-synthesis method for morphing between two timbres of a source and target sounds. The morphing is done using the Y axis and the continuous pitch is derived from either mapping the X-position either directly or using MIDI note with 14bit pitch bend. Other parameters such as Size is used as an alternative Z control, albeit with a side-effect since it is at the same time affecting the Y position. The continuous pitch controller paradigm is limited in terms of range and playing surface, and therefore has very limited use to explore polyphonic continuous pitch. However, despite such constraints the combination of timbre morphing and continuous pitch even confined to use it monophonically provides an expressive control of sound material. A temporal exploration of sound and finger motion is done using *at-granular* which features four granular engines - each corresponding to four fingers. It features a mapping of finger motion data to grain triggering parameters as opposed to a more traditional use where grain parameters are controlled often using random or periodic functions, making way for an engaging activity. Although the action-sound relationship is more unclear and dependable on

the selected sound, the idea of motion as temporal control of sound is explored with artistically interesting sound-producing results. A small experiment was also developed, the *at-noise*. This instrument uses a noise oscillator which is fed through a State Variable filter. Controlling the filter parameters such as frequency and the Q-factor, as well as the amplitude using multi-touch. The instrument makes for a very simple sound of motion exploration where the magnitude of finger motion is a contributing factor.

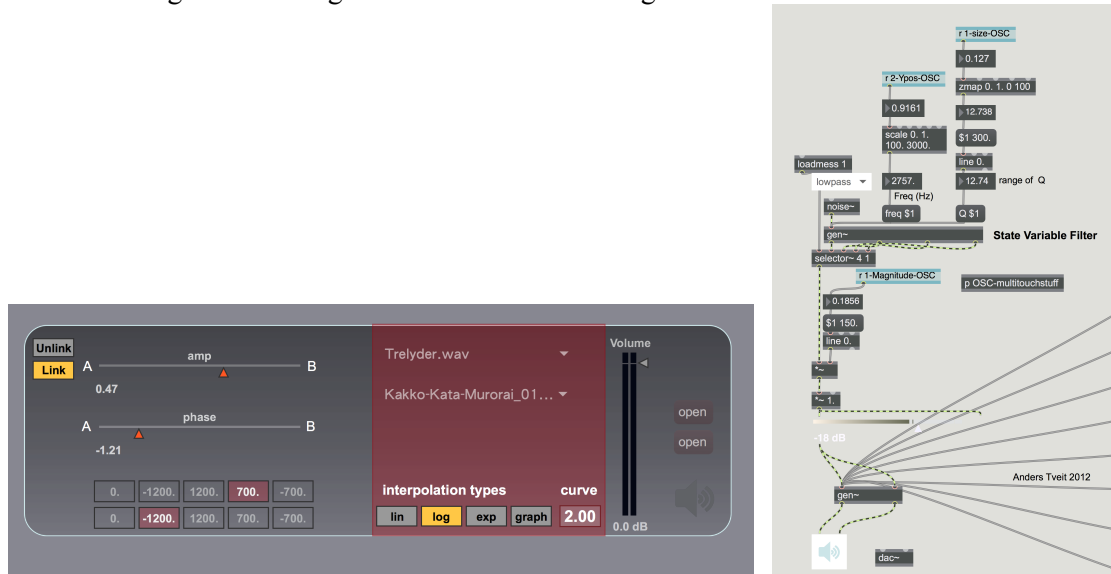


Figure 4.4. Left: The *at-crossConvGRAN Hybrid*. Right: Shows the inner working and mapping of *at-noise*. N.B. The last *gen~* object at the bottom in the figure is a standard hall type reverb with pre-set parameters.

4.5 Lights, Camera, Action?

The laptop's native physical input capabilities consist of more than just the keyboard and the trackpad. On MacBook models by Apple one finds a high resolution camera, light sensor, and accelerometer. The camera and light sensor have been explored by various Laptop Orchestras and various media and sound artists. However, after experimenting with these interfaces in my artistic practice, and in relation to this thesis, I have found them quite problematic. That is not to state that they do not provide any value or possible interesting artistic use. In-fact I have explored and used them at times for sound installation purposes, or in conjunction with

other instruments for controlling various parameters in known-conditions. And I underline *known-conditions* because one of the problems with such devices is that they are highly dependent on environmental factors. Environmental factors include: variable lighting on the stage, occlusion - objects in way of the lenses or shadowing of the image etc. This requires calibrations each time there are changes in the environment. And to account for such changes in real-life-on stage situations are highly impractical, especially in music making and improvisation. While it may be possible to mitigate such environmental issues through the use of, for example, machine learning, (Chandel 2015) it would pose a major latency issue for any real-time use and thus greatly affect a performance. Another issue is that such devices do not provide any clear physical boundary nor tactile feedback. Which in turn makes it hard to define ranges and provide 'feel'. While this is also true for "tried and tested" instruments like the Theremin, for me, the sum of these problematic issues and the lack of touch i.e. tactile feedback, fails to provide the engaging activity I had as a criterion for using the Laptop as an instrument. Therefore, I have determined to not include these forms of input in this thesis for further discussion of use, choosing only to briefly mention one here in this section as examples of early experimentation; The *light-sensor*. The *light-sensor* uses the built-in light sensor for tracking changes in ambient lighting, but also changes to the LCD backlit keyboard function. These changes are reported back and the data is normalised and sent out as either OSC or MIDI. Two uses which I did experiment with was to use the change of ambient light as an induction to the amount of jitter or parameter variation control to a granular synthesis engine. And, by blocking and releasing the sensor input (located at the built-in camera) one could use this as a parameter pitch bend. Again, this works well, provided the conditions of the environment are known. The sensor is also quite slow in its update rate (200 ms.), so any fast changes would not transpire.

4.5.1 The Smack Machine

One native input sensor which to some extent has a physical boundary, is the SMS sensor. The Sudden Motion Sensor or SMS is a built-in triaxis accelerometer sensor originally used in Laptops as “Active Hard-Drive Protection” A technology that alerts the computer system when acceleration and vibration is detected and protects the hard-drive from failure. By accessing the sensor data one can map the data to sound-producing-parameters, for example by using tilt gestures or, as in my creation the *Smack machine*, by tapping or hitting (lightly!) on definable zones on the laptop chassis. It consists of two Max abstractions: *at-motion* and *at-smack*. The *Smack Machine* uses the aka.bookmotion third-party external for Max by Masayuki Akamatsu <http://www.iamas.ac.jp/~aka/max/> which is based on the Unimotion library by Lincoln Ramsay <http://unimotion.sourceforge.net/> allowing access to the built-in Sudden Motion Sensor. The Max abstraction polls the sensor at a given rate (default 10 ms) to output data on the x, y, z axis. *at-smack* calibrates the accelerometer and then outputs the normalised, filtered and scaled accelerometer values to OSC for communication. The abstraction takes five control parameters: *On*, *calibrate*, *samplerate*, *threshLo* and *threshHi*. It will also return the current state of the control parameters if the message *getState* is send. The Smack-machine sends out four, what I describe as, event parameters: */hit*, */vel*, */bottom* and */sides*. The *threshLo* parameter sets the lowest threshold of detection of a hit – it is, by default, set to 10. Higher values than 100 are impractical. The *threshLo* parameter must be set in accordance to the *threshHi* and *smoothsize* parameter. The *threshHi* parameter sets the higher range of the detection and affects the desired top value of velocity. The *smoothsize* parameter is essentially the amount of lowpass filtering effectuated on the sensor output. (1.-0.01) The Threshold and *smoothsize* parameters is dependent on each other and will affect the overall sensitivity. OSC namespace and event parameters from: IP address localhost @ port 7401: */at-motion/smack /hit(0-1.) /vel(0-1.) /top(0-1.) /bottom (0-1), /sides(0-1.) /returnState (s,l)*. The Smack Machine, when being hit, detects two different zones for where the hit happened on the laptop chassis: At the bottom/top or at the side. These detections can be mapped to, for example, a percussive physical model synthesis which can create some interesting results and uses. Other uses might be to use it to trigger samples. However, a

major drawback is that it requires calibration, and given its sensitivity even when correct thresholds have been set, it might trigger unexpectedly as it picks up vibrations. So, in this too, environmental factors and conditions apply. Environmental issues can be mitigated by placing the laptop on a material that will dampen vibrations, but it will nevertheless require special attention. In addition, such issues and for reasons that were out of my control, Apple have since I started developing the SmackMachine changed the technology. The sensor updates at much lower rates or in some cases the sensor has been removed on newer models which renders this laptop instrument unusable for current and future laptops. But, despite this, I decided to include this laptop instrument as a reference for my work in this thesis.

Chapter 5

5 Evaluation

Reflections, discussion, and 'evaluation' based on experiences from the development and artistic work. Further enhanced by recorded documentation.

5.1 Performances, conditions, and the field

In addition to the important work of experimenting and exploring the application of these instruments during various stages of development, the laptop instruments were used in a range of performances with groups and ensembles of different sizes and instrumentation. This included quite a vast range of performance situations, from regular concert performances, rehearsals, and studio recordings, to more informal ad-hoc performances.

From this I have selected the following recorded examples and the corresponding filenames to be included in the thesis:

Trio: (Grand Piano, Percussion and Laptop/Electronics).

Ex-1_MultiTouchContPitch+at-noise.mp3 (audio)

Ex-2_MultiTouchGranular.mp3 (audio)

Duo1: (Drums and Laptop).

Ex-3_MultiTouchGranular+KeyboardPolyPoly.mp4 (video)

Trio: (Grand Piano, Percussion and Laptop/Electronics).

Ex-4_MultiTouchGranularCross+FXs.mp4 (video)

Ex-5_MultiTouchMappingFXs.mp3 (audio)

Ex-6_Keyboard_AlfaBetaRedux(early_version).mp3 (audio)

Quartet: (Percussion, Guitar(prepared), Percussion and Laptop/Electronics)

Ex-7a_KeyboardAlfaBeta_Redux.mp3 (audio)

Ex-7b_KeyboardAlfaBeta_Redux.mp3 (audio)

Trondheim Jazz Orchestra: (22 piece Big-band including two vocalists.)

Ex-8a_KeyboardPolyPoly(early_version).mp4 (video)

Ex-8b_KeyboardPolyPoly(early_version).mp3 (audio)

Duo1: (Drums and Laptop).

Ex-9_KeyboardPolyPoly.mp4 (video)

Duo2: (Electronics and Laptop)

Ex-10_SmackMachine.mp4 (video)

The performances capture aspects of how I have used the laptop instruments in the various constellations and expressions where the common denominator was an improvised music context. This gave me a solid basis for using them in a range of varied conditions. However, in general, the impetus for the performances with a group/ensemble did not come from an intent to do a specific evaluation for this thesis. The instruments, when used in such performances, were developed as a part of my artistic work in the situations I felt the use of the laptop instruments would either be interesting to experiment with and/or artistically fulfilling. At times the experimentation with different instruments showed that they did not work well in the context of what I determined to be either desirable in terms of sonic result, or in terms of technical and practical factors. However, this was more prominent in the early stages of development and experimentation. Since the start of this "journey" the instruments have been through a number of iterations and refinements, a process of evolution based on both my gained experience of performative use and accumulated technical understanding. However, it is important to underline that they are by no means at an ideal or "frozen" state, in fact they are still evolving, details are being scrutinised and changed without changing the overall functioning. This also applies to other sound producing software I develop and use. Such a dynamic relationship with a digitally designed instrument and its perpetually evolving state is perhaps the norm rather than the exception, where digital music instruments, due to their arbitrary nature, rarely reach their final designs (Magnusson 2010).

5.2 On evaluation.

Although a range of frameworks and guidelines has been proposed for evaluating digital music instruments (O'Modhrain 2011), they are mostly derived from the field of human computer interaction based on quantitative measurements. A quantitative method of evaluation is difficult in this thesis, given the complex nature of musical experience and the difficulty of quantifying the components of such activities. Additionally, questionnaires or forms were not utilised. It is therefore important to ask what does evaluation mean in this context? For me as the performer, the evaluation has everything to do with my experience of performing with these instruments and how well I think they worked in the range of situations and conditions I used them in. But I am, at the same time, the developer/designer of these instruments and also a member of an ensemble doing improvised music. This blurring of roles poses challenges regarding evaluation as it is difficult to evaluate only from one perspective, since the different roles impact each other. For example, if during or after a performance I was not pleased with the outcome, it could be hard to assess if it was something related to my implementation or the design of the instrument which impacted on my performance and/ or the collective ensemble performance. Was it just me having a bad day as a performer? Was it due to a collective lack of effort, or collective decisions that did not lead to the desired direction in the improvisation? From a developer's perspective, such issues could mean an improvement is needed - I might change the mapping or a function, potentially influencing the next performance. On the other hand, a technical flaw rendering the instrument or a function unusable, could also bring out a creative solution from me as a performer, which in turn can become the driving force for the ensemble. Therefore, a consideration of the different roles or stakeholders (ibid) involved in the use, development, and the perception of the laptop instruments can be useful when evaluating. Since in this case all the stakeholders are me, with the exception of the other various members of the ensembles, it is most fruitful to use a broad definition of evaluation: An evaluation consisting of the discussion and reflection of the process as a whole, taking all the different stakeholder

perspectives.

The evaluation is structured by grouping each instrument as described in Chapter 4 by its input interface.

5.3 The Multi-Touch instruments

As described in Chapter 4.3.2, one of the instruments I developed was for using the Multi-Touch trackpad as a continuous pitch control with morphing of timbre. The main reason for this was not necessarily to be able to play melodies with different timbres as such, but for the possibility of a continuous transposition and morphing of any sampled material. I wanted to explore this instrument during an upcoming live-in-studio recording session with the improvisation trio *pd conception*. A trio consisting of Grand Piano, Percussion and myself on electronics / laptop. As there already was a pitched based instrument in this trio, I was interested in exploring how I could use pitched material or alluring to pitch in such a setting. Perhaps giving the pianist another option of counterpoint in the interplay or giving him freedom to explore other roles outside of pitch. One of the challenges of having a continuous pitch-based control against a fixed pitch instrument like the piano is the level of precision required if playing in tune with the piano is desired. This was especially difficult considering the playing surface of the trackpad is quite small. The attraction of continuous pitch is having the freedom to explore microtonality, so I did not want to give up such a possibility. But at the same time, I still wanted to be able to perform easily in tune with the piano. To solve this, I developed a rounding function in the *at-multitouch* where the starting note is instantly rounded off to the nearest equal tempered pitch, but the rounding is released when sliding the finger up or down in a continuous fashion. The performance example

Ex-1_MultiTouchContPitch+at-noise.mp3 is from the track *So Long Mishka* taken from the live-recording session of the album *Baikonur* (*pdconception* 2013) and illustrates how I used the *at-crossConv+Sampler* instrument for continuous pitch and sound morphing. One of the aesthetics behind this trio is creating a "sound" which ambiguates the difference between

what is electronic and acoustic in order to create a certain organic quality. I wanted to explore this ambiguity by the use of concrete sounds while at the same time alluding to an instrumental type sound. Using two concrete sounds derived from the same material, where the sound of clinking two crystal glasses provides an iterative type of sound and the other a crystal glass being struck giving a defined sustained pitch. By morphing between those two sounds and using the continuous pitch transposition I was able to create both a haunting effect and provide a lead-voice during the improvisation. In this improvisation I also made use of the *at-noise* instrument which is a noise oscillator running through a filter controlled by the Multi Touch trackpad. This was used in a shorter section (04:00 - 05:10) for creating an ocean wave like atmosphere. An interesting point is the interplay between the percussion and the continuous pitched glass sound from laptop during the first section of this track, approximately: 0:40 - 3:45. The percussionist plays a metal bowl filled with water, where he changes the pitch by playing on different parts of the bowl. Where we establish a sort of duet following each other, complimenting the pitched inflections in turns. As a laptop performer this gave me such enjoyment, being able to having a more direct control of sound – to truly focus on listening, responding immediately, being present in a "now" moment. For the other performers this might also had an impact in that they did not need to adapt their playing or "wait for something to happen in the electronics". While the use of the *at-crossConv* and a sampler for continuous pitch and morphing created an eye opener in this context I was at times annoyed with the limited surface area of the trackpad. Being limited to a transposition of 10 semitones with manually having to transpose octaves compromised the potential of the instrument a bit too much. The *at-noise* as used here in this performance proved to be ample for creating an atmospheric texture, despite being a bit one-dimensional in terms of the musical expression on a larger scale. It should be noted that there is a dynamic rhythmic element in the form of a long loop in the background nearly throughout the whole track. This stems from a discussion we had within the ensemble about the use of such loops or drones for creating a foundation for acoustic performers, in their words: "to be able to play on" as this was something they were used to doing with other laptop/ electronic performers. However interesting it is, it is not within the scope of this thesis to discuss such cultural expectations.

During the recording session with the same ensemble I also used the *at-granular* for exploring how granular synthesis could be used in quasi-rhythmical fashion and using multiple granular sources. The performance example *Ex-2_MultiTouchGranular.mp3* is also a performance taken from the same album (ibid). Again, a concrete sound was used, where the sound of clinking stones/gravel had been loaded into the first grain module of *at-granular*. Into the second grain module a live sampled tom-tom phrase from the percussion had been recorded. The live sampled sound was sampled during the end of a previous improvisation in the session. The mapping of finger motion to the *at-granular* was following the same scheme as described in chapter 4.3.3:58. Although the implementation of the granular engines in *at-granular* as used in this performance was made using the FTM library by IRCAM. The difference from the current version of *at-granular* which use the MUBU toolbox by IRCAM is minuscule. The performance starts with shaping the gravel sound, creating on / off phrasing together with the percussionist and the pianist which at this point were using the CrackleBox from STEIM (a simple analogue conductive synth making crackling noises) before he uses the body of the grand piano as a percussive instrument. The use of the live sampled tom-tom phrase enters at 0:34 together with the gravel sound which continue at high grain frequency and short grain duration. Even though it's impossible to decode any singular meaning from a performance in a group improvisation as such as this, it seems the whole improvisation developed and revolved around the idea of creating these fragmented rhythmic phrasings. Where we collectively are playing on and off phrasings using only percussive sounds creating this ambiguity of source sound and cause. A strong point for using granular synthesis is the temporal shaping and texture it provides, but it gets really interesting for a performer when shaped by motion. And for me, it lends itself quite well to the type of sound exploration and interaction as employed in this performance. Using Multi-Touch motion for controlling granular synthesis gives the direct feeling of the very shaping of a sound. But because it is dependent on the sample material it has an unclear action-sound relationship. Reflecting back on this performance, it involved an element of risk-taking when I live-sampled the percussion because there are uncertainties of the behaviour of the transformed sound when performing using un-tried material with the *at-granular*. However, such risk-taking requires me as a performer to rely on my skills, and forces an active engagement with the sound and the

musical environment.

Another use of the at-granular has been in combination with other laptop instruments. In the following performance example: *Ex-3_MultiTouchGranular+KeyboardPolyPoly.mp4* it is used to create iterative → sustained textures and for creating various noisy and dynamic shapes in an interplay with both the drums and the percussive sounds from the at-PolyPoly instrument. The performance in question was an Ad hoc performance that took place at a rehearsal space in a duo with a drummer. The core idea I had for this ad-hoc ‘performance’ was to explore how to use the laptop instruments for a very direct action to sound, as a percussive instrument in order to establish a tight interplay with the drums. While the laptop keyboard instrument, the at-PolyPoly relies on the intrinsic dynamics in the samples, the at-granular can be shaped in-place for creating a more varied dynamic expression. Three various samples were pre-loaded into the respective granular modules, where the first module corresponds to the first finger on surface, the index finger as featured in the video example (0:00 - 0:40). Where is used for creating an iterative towards sustained softer sound. The use of multiple fingers, up to three in this case is not used for any refined control of the grain parameters but for creating dynamic bursts or more noisy dynamic shapes of sound, such as used in 1:20 - 2:32, and 3:14 - 3:50 in the performance example.

Of the 16 features (Chapter 4.3.1) that the at-multitouch extracts per finger from the Multi-Touch, between 5-8 of them had been used in the aforementioned instruments and performance examples for mapping. Two performance examples that use a more extensive set of these features, are examples where the *at-crossConvGRAN* together with a mapping of motion features to other DSP effects is used. The *at-crossConvGRAN* is a hybrid between the *at-granular* and the *at-crossConv*, where two granular engines taken from the *at-granular* can be cross synthesised as respectively source A and target B. The first example is taken from a concert in 2013 with the aforementioned trio. In *Ex-4_MultiTouchGranularCross+FXs.mp4* the pianist is using his left hand inside the soundboard of the grand piano and the right hand for playing a rhythmical ostinato where the drummer/percussionist adds compliments using the floor tom drum. I perform a small solo

section on this backdrop (0:00 - 0:40) using the *at-crossConvGRAN* with a pre-loaded sample at the source A. Inside the grand piano soundboard I had attached a contact microphone for picking up sounds made on the strings and the soundboard. I then live-sample/ continuously record the audio from this microphone into the *at-crossConvGRAN*, second target B granular module. In addition to this the *at-crossConvGRAN* output is running through a series of DSP effects such as filters, delays and tremolo. The DSP effects parameters are controlled by a number of features such as acceleration, kinetic energy and finger angle derived from the Multi-Touch using the *at-multitouch* (4.3.1). However, the DSP effects are only activated when the phase/amp controlled by the y-position of the second finger is 35% towards the target B sound of the *at-crossConvGRAN*. The live-sampling, the effects and morphing between the source A and B granulation can be heard from 0:50 - 2:20. The small mixer which can be seen in the video is used for adjusting the gain and level from the contact mic into the input of the *at-crossConvGRAN*. These multiple features - multiple audio processes mapping as used in this performance can then be best described as various combinations of one-to-one, one-to-many, and many-to-many (Lazzetta in Miranda, Wanderley 2006). A perhaps clearer example of these multiple features - multiple audio processes is heard in *Ex-5_MultiTouchMappingFXs.mp3* where two pre-loaded audio samples are used for granulation and morphing. A range of DSP effects of similar types to the previous performance example is used consisting of Delays, Filters, Tremolo. The mapping here is also similar to the previous example, where a range of features are mapped to the various parameters of the DSP effects, the granulation modules and the morphing parameter. As in both of the examples the use of additional effects does provide for me, as a performer an extra layer of shaping the sound further. However, deciding on which effects and their possible behaviour (both technical and sonically) in such a setting, a crucial aspect to this is to set aside enough time for consideration and experimentation. Although I have not explored the use of Multi-Touch for only controlling effects other than in conjunction with my aforementioned laptop instruments, it is nevertheless a viable strategy to use and explore.

5.4 The keyboard variations

Although the use of the keyboard as a controller started with the $\alpha\beta$ (4.2) originally as a part of the composition of the same title, it was when I started experimenting with it as a live-sampling instrument that it gradually evolved into the two variations $\alpha\beta_Redux$ and $\alpha\beta_PolyPoly$. Even though the two instruments share a common root and are used percussively, they are quite different in terms of technical functionality and how I use them in performance. Since they developed over time I have included some performance examples that also underline this evolutionary process of development. One of the first uses of the $\alpha\beta$ as live instrument was, again, with the pdconception trio. During a soundcheck I wanted to test how this could work as a live-sampling instrument

Ex-6_Keyboard_AlfaBetaRedux(early_version).mp3 using only the internal mic from the laptop. The example is just the raw output from the instrument itself and does not provide the whole ‘picture’ but it gives an indication on use and how the word-triggered effects are being deployed at various points. The discovery of using this as a fast way of live sampling eventually led me to develop the $\alpha\beta$ into the $\alpha\beta_Redux$ where the sampler function was greatly improved and the effects were changed to be more aligned with the type of processing I tend to do in live performances. A designed constraint of this instrument is that it can be difficult to use on sustained sounds, and was better suited to short impulse / percussive type sounds. Therefore, I tend to live-sample percussion instruments - or as in the following example use it for live-sampling a prepared guitar. *Ex-7a_KeyboardAlfaBeta_Redux.mp3* is from a performance and collaboration I had with *Parallax*, a Norwegian improvisation ensemble recorded during a tour in the US. The example is from a duo section which features myself on laptop using the $\alpha\beta_Redux$ and the guitarist using prepared guitar and extended techniques. The brief example showcases the percussive sonic outcome of the interplay but also the instantness of the live sampling, with fast-changing transposition possible. A reverb with fixed settings is also added to the output of the $\alpha\beta_Redux$, this is done for two reasons. Firstly it is used as a way to sustain the shortest sampled fragments slightly, which can be very

short due to the possible max pitch/speed ratio 16va (3:1 / +24). Secondly it is used to add some depth, however some considerations to the reverb setting is to be taken dependant on the type of the acoustic instrument or material. As a performer using the *αβ_Redux* one is dependent on what the other instrument(s) are producing sound-wise in order to perform, creating the impression that the instruments are fusing into one instrument. This is something that can be quite interesting sonically, therefore a crucial aspect to any successful use of the *αβ_Redux* is to get the input level and gain stage correct - with as little noise in the signal chain as possible as any unwanted noise would be transposed, creating an undesirable effect. In this example the guitarist had a high-quality piezo microphone mounted on his guitar with a proper line level output running to into my system. Another similar use with the same quartet is in another performance where I use it on percussion

Ex-7b_KeyboardAlfaBeta_Redux.mp3 In this particular example there is a lot of other things happening at the same time coming from the other acoustic instruments as well, but the various fast-changing transpositions can be heard quite well.

An early version of what was later to become *αβ_PolyPoly* was developed for use in two sections of the piece *Night, Shattered Sky* by Norwegian Jazz composer and musician Øyvind Brække written for the Trondheim Jazz Orchestra (Brække 2011).

The first section is an open improvisation which features myself on laptop, two female vocalists, (alto and mezzo-soprano), drums/percussion and the baritone saxes entering at conducted points in time. In the brief live-in-concert extract taken from this section,

Ex-8a_KeyboardPolyPoly(early_version).mp4 I am using various short shattered glass samples and synthesised percussive sounds triggered by using the built-in laptop keyboard. And similarly, from the studio recorded version of the piece, the second section features the rest of the orchestra performing written parts: *Ex-8b_KeyboardPolyPoly(early_version).mp3* Despite this very simple use as heard in these two examples, it proved to be very effective and direct way of interaction, where I could react swiftly to the other performers' phrasing and lines. I later developed the concept of fusing ideas from the *αβ_Redux*. Despite the fast and effective way of live-sampling that the *αβ_Redux* affords, as a performer one is dependent on other instruments in order to make sound – though this can produce highly

interesting results with focused and active engagement. The preparation of setting of mics and getting ample signals into to the system can sometimes be a time-consuming process. The need to provide an alternative whilst retaining the use of the keyboard in a direct action-sound approach along with the experience from the work I did with the *Trondheim Jazz Orchestra*, led me to develop the *αβ_PolyPoly*. As a laptop/electronics performer I quite often like to think as a percussionist - where one has access to a range of different instruments and materials at hand. As described in 4.1.3 the *αβ_PolyPoly* uses an embedded sample bank of 138 percussive sounds as the sounding material, triggered in various ways through filters and the word-based effects as used in *αβ_Redux*. This allows me to be quite rhythmical and affords to establish a tighter interplay. The recent version of *αβ_PolyPoly* as used in *Ex-9_KeyboardPolyPoly.mp4* illustrates the various concepts and possibilities for this type of interaction. The use of the word-based audio effect concept is illustrated at 1:41 - 2:05 creating an aggregate functioning to the sound. Performing using these effects do add an extra challenge in order to execute the desired effect. So, from a performer's perspective, considerations as to the length and complexity of the word needs to be done where the focus should be on the sonic outcome and not be a cognitive disturbance. In this example I use a simple word of few characters which also fits the sonic components which I define on the spot at: 1:37 - 1:41. One of the designed constraints in *αβ_PolyPoly* is that the dynamics are intrinsically linked to the samples, but by being familiarised with the sounds and the various functions of the instrument one can, for instance, create accented sequences and motifs as illustrated in the example 2:54 - 3:46. The instrument is very simple in its use as it has a highly familiar interface although abstracted in the location of it sounds, but as a performer I do take joy in using it for percussive roles, especially in a particular musical setting as in this performance.

5.5 The Smack Machine

The Smack Machine is perhaps the most novel of the laptop instruments. It uses the built-in accelerometer sensor for detecting hits on different places on the laptop chassis. This is done through various filtering of data and threshold detection of impact, where the core idea was to use this as kind of a drum. In the sense of its potential for musical expression it has very clear constraints and limited use due to practical issues as described in 4.5.1. Furthermore, given the change in technology (4.5.1 para 2) unfortunately, it does not provide any real practical value for future development or performances other than a reference to its historic use. As a reference to the few performances I did using the Smack Machine, one example is during a duo performance *Ex-10_SmackMachine.mp4*. In this performance I had mapped several sounds to be triggered during a registered hit, depending on where on the laptop it would trigger a particular group or constellation of sound. Also depending on the velocity of the hit it would also select the number of sounds that constituted the constellation. The video example shows this interaction, where the other performer provides the musical basis.

Towards the end of the example it also illustrates an issue of the instrument, an issue of its detection threshold (1:09 - 1:16). Here I have clear problems triggering a sound as finding the sweet-spot of detection threshold is quite difficult. Setting the threshold for detection any lower, which could have helped in this case could also be detrimental as an acoustic change in the surroundings such as through acoustic low frequency vibration would have an impact on the detection. Although it has a certain theatrical flair to its use, the sum of issues surrounding the instrument outweigh the performative gains.

5.6 Remarks on the evaluation

The goal of this evaluation has been to provide a broader understanding of the instruments and their various qualities as tested through development and artistic use. Where both the development and artistic practice was done over longer period of time. The artistic use was in a variety of situations and in constellation with other performers. The evaluation took its form through discussion and reflection of the process as a whole, looking at different perspectives from the use, development, and perception. From this, a picture emerges: The laptop instruments tend to be used for specific purposes even in various contexts, their use is cultivated in specific sonic qualities. The interfaces used for interaction are familiar through their everyday use which, from the performer perspective, is a strong point. But the physical constraints of the interface, as in the use of Multi-Touch trackpad for pitch control, limit the number of polyphonic voices and pitch range. Or in the case of The Smack Machine they can be dependent on the native technology and environmental conditions. The instruments have an instantness in the action-sound interaction which provides both the performer and fellow performers to react and interact to instant decisions. Which, on the basis this of evaluation, seems to be an important factor for engagement and interplay. While the action-sound relationship is unclear in some instruments such as in the at-granular, its overall shaping of sound by finger motion is the pertinent and performative factor. The word-based audio effect design of the $\alpha\beta$ range of instruments are considered a novel design and do add challenge but considerations and a balancing for potential cognitive disturbance must be addressed. The developed instruments have on occasion been used in combination in performances, but mostly used singularly. The instruments have gone through various iterations with improvements or slight changes of functionalities. Some, as in the case of the continuously pitch control had changes in order to alleviate a specific performance problem (Pitch rounding). One specific instrument, the $\alpha\beta$ _Redux relies on other performers sounds to be able to produce sound itself. This can be seen as an interesting notion of a new instrument, where the acoustic source instrument and the live-sampling instrument seems fused, creating a focused engagement in the sonic attention and interplay. But it may have practical issues

where one is relying on recording devices and other hardware for successful use. Other instruments have live-sampling as an option where a combination of live-sampling and pre-existing sound material and their transformations can constitute the instrument's sound. While live-sampling provides a flexibility it is also a precarious undertaking as the behaviour of the transformed sound can be uncertain/unknown. On the other hand, it relies on the skills of the performer and an active engagement with the sound and the musical environment.

Chapter 6

Provides a short summary of the thesis, reflections on the process of the project and areas of interest for further research.

Concluding Remarks

This thesis developed from my interest in using the laptop within the context of improvised music, and the challenges that arise from the laptop's constrained native physical interfaces for interaction. The focal point for this interest was located in the development a range of laptop instruments where the interaction was constrained to these interfaces. And importantly how the instruments could be designed and used to facilitate an engaging activity in the interactive relationship in an improvised music context. An activity where I, as a performer, am able to focus, react to and interact with instant actions carried out in the musical environment. I have found, through the evaluation of development and performances done in a variety of improvisational contexts, that the instruments based on two of the interfaces facilitate such engaged activity. Where the instruments' physical and designed constraints, and the instantness to action-sound is a contributing factor. That is not to say they are without issues as there are technical dependencies and considerations to be made. By looking closer at four paradigms for live performance activity and the historical development in technology, I have also highlighted issues and aspects of using the computer as a performance instrument. The design and development of laptop instruments was also informed by theoretical studies into digital musical instrument design, where the concepts of affordance and constraint are central to the design and its creative use.

Improvisation

"Sounds are placed: placed in contrast to, in parallel to, in imitation of, in respect of, without regard to, other sounds. Minds struggle, coalesce, defer or acquiesce. Inner debate meets outer debate. Instant decisions dictate the immediate direction of the music." Eddie Prévost (Prévost 1995:3).

I have touched upon the subject of improvisation in relation to this thesis, and while it might seem strange to not go deeply into this as it is part of the field in which this thesis has taken place, it has been a conscious decision not to do so based on the scope of the field of inquiry. In order to provide a broader picture, improvisation should be looked at from various perspectives and angles which would include research fields from cognitive, sociological, neuroscience, musicological sciences, and artistic research, to name a few. There are already in this thesis a variety of topics and subjects that crossover from a number of different fields. And to be venturing into even more would require a lot more to this thesis scope.

Audio as an interface

Other than the explicit use for input and output of sound, one aspect of the built-in physical interfaces which has not been mentioned as a potential use for direct interaction is the audio interface. This is largely due to the complexity and the scope of such use where, for instance, it would require looking deeper into research fields such as Music Information Retrieval (MIR), Machine learning and Digital Signal Processing (DSP). But also because I could potentially see some pitfalls or issues in connection to this where increased DSP calculation cost in conjunction with the sound-producing generation could cause stability problems and the possible increased latency directly affecting performance. However, these topics and fields of research are something I am currently highly interested in, and parts of this will be considered in future work.

What comes next

The development of these instruments is continuously changing and evolving. But there are other areas of interest as mentioned in the section above. For example, gesture recognition with Multi-Touch trackpad as already used natively in the OS is an area of interest. I am interested in exploring this in the same way as the word-based-effects as used in the Keyboard based *αβ* range of instruments. In related areas I already have made some development and experimentation which has shown some promise into what I would like to explore in terms of using feature extraction of audio. In particular to integrate feature extraction from the surrounding musical environment for automatic selection of subsets of the existing sample-bank into the *αβ_PolyPoly* instrument. But the question of adequate response time and latency is a key factor. Other ideas to pursue is by the use of convolution or by Linear Predictive coding techniques (LPC) for fusing the live sampling capabilities of the *αβ_Redux* with the sample-bank based approach from *αβ_PolyPoly*. Where the samples frequency response or the spectral envelope are dependent on the technique, would be imposed onto the live-sampled sound. The MPE (Multi Polyphonic Expression) MIDI standard is also something which would make MIDI conversion easier when it comes to the at-multitouch. Although it has not been mentioned in the thesis, I am very aware that physical modelling synthesis can be a viable and powerful option and certainly a very shapeable sound generator for the kind of interaction I have explored in this thesis. But at this point I have not yet been aesthetically content with the sonic quality of such techniques fitting into my artistic practice. However, I will keep the option open for any change in the future.

Bibliography

- Bin, S.M.A., Bryan-Kinns, N. and McPherson, A. (2018) "Risky business: Disfluency as a design strategy". *New Interfaces for Musical Expression (NIME)* Blacksburg VA 2018.
- Brække, Ø. (2011). "Night, Shattered Sky". *Migrations, Trondheim Jazz Orchestra & Øyvind Brække*. © MNJ Records MNJCD 2011.
- Cascone, K. (2000). "The Aesthetics of Failure: "Post-Digital" Tendencies in Contemporary Computer Music" In *Computer Music Journal* 2000 24 (4), pp. 12-18
- Cascone, K. (2003). "Grain, sequence, system: Three levels of reception in the performance of laptop music" in *Contemporary Music Review* 22(4), pp.101-104
- Casserley, L. (2001). "Plus ça Change:" Journeys, Instruments and Networks, 1966-2000. *Leonardo Music Journal*, 11, 43-49.
- Chadabe, J. (1977). "Some Reflections on the Nature of the Landscape within Which Computer Music Systems are Designed" *Computer Music Journal*, 1(3) pp 5-11.
- Chadabe, J. (1980). "Rhythms, -For Computer and Percussion". Vital Records 1981
- Chadabe, J. (1984). "Follow Me Softly" CDCM Centaur CRC 2310, CDCM CE130.
- Chadabe, J. (1997). *Electric sound: the past and promise of electronic music*. Upper Saddle River, N.J: Prentice Hall.
- Chandel, H., & Vatta, S. (2015). "Occlusion Detection and Handling: A Review." *International Journal of Computer Applications* 120(10):33-38, June 2015.
- Collins, N. (1981). *Little Spiders* on Nicolas Collins & Ron Kuivila, *Going Out With Slow Smoke*, Lovely Music LP, 1982.

- Collins, N. (2007). "Live Electronic Music." In *The Cambridge Companion to Electronic Music*, edited by Nick Collins and Julio d'Escriván, pp. 38–54. Cambridge Companions to Music. Cambridge: Cambridge University Press.
- Collins, N. (2009). "Before Apple There Was Kim – the Microcomputer, Music and Me" Online Available <http://www.nicolascollins.com/texts/microcomputermusic.pdf>
- Collins, N. , McLean, A., Rohruber, J., Ward, A. (2004). "Live coding in laptop performance" In *Organised Sound* 8 (03), pp. 321-329
- Dean, R. T. (2003). *Hyperimprovisation: computer-interactive sound improvisation*. Middleton, Wis: A-R Editions.
- Di Giugno, G, & Armand, J. P. (1991). "Studio Report IRIS" In *Proceedings of the IX Colloquio di Informatica Musicale*, Genoa, Italy, pp.358-62.
- Emmerson.S. (2007). *Living Electronic Music*.115:116, Hampshire and Burlington: Ashgate, 2007.
- Essl, G. Rohs, M. (2009). "Interactivity for Mobile Music-Making" In *Organised Sound* 14(2), pp. 197–207
- Fiebrink, R., Wang, G. and Cook, P.R. (2007). "Don't forget the laptop: Using native input capabilities for expressive musical control." *Proceedings of New Interfaces for Musical Expression (NIME)*, New York City, USA, 2007.
- Fiebrink, R., Trueman, D. Cook, P.R. (2009). "A meta-instrument for interactive, on-the-fly machine learning." *Proceedings of New Interfaces for Musical Expression (NIME)*, Pittsburgh 2009.
- Gibson, J. J. (1979). *The Ecological Approach to Visual Perception*. Houghton Mifflin.
- Grossman, R. (2008). "The tip of the iceberg- laptop music and the information- technological transformation of music" In *Organised Sound* 13(1), pp. 5–11 2008 Cambridge University Press

- Gustavsen, T. (1998). *"Improvisasjonens dialektiske utfordringer."* Thesis, University of Oslo.
- Hoelzl et.al (2006) "PowerBooks_UnPlugged" Online. Available: <https://web.archive.org/web/20060827053223/http://pbup.goto10.org/>
- Holmes, T. (2008). *Electronic and experimental music: technology, music, and culture*. New York, NY ; Routledge.
- Irving, M. (2006). "Interpretive Practice in Laptop Music Performance" Master Thesis, Nova Scotia College of Arts & Design.
- Jensenius, A.R. (2007). "ACTION -- SOUND - Developing Methods and Tools to Study Music-Related Body Movement". Ph.D. Thesis, University of Oslo.
- Jordà, S. (2005). "Digital Lutherie: Crafting Musical Computers for New Musics' Performance and Improvisation". Ph.D. Thesis, University of Pompeu Fabra.
- Jordà, S. (2007). "Interactivity and Live Computer Music" In *The Cambridge Companion to Electronic Music*, edited by Nick Collins and Julio d'Escriván, pp. 86–103. Cambridge Companions to Music. Cambridge: Cambridge University Press.
- Kaltenbrunner, M., Jordà, S., Geiger, G., Alonso, M. (2006). "The reactable: A collaborative musical instrument" In *Enabling Technologies: Infrastructure for Collaborative Enterprises*, 2006.
- Kirn, P. (2011). *Keyboard Presents the Evolution of Electronic Dance Music*. Backbeat Books
- Lindborg, P.M. (2006) "Reflections on Aspects of Music Interactivity in Performance Situations" In eContact!10.4.
- Lyon, E. (2002). "Dartmouth symposium on the future of computer music software: A panel discussion" In *Computer Music Journal* 26 (4) 2002.

- Luta.P. (2009). "Take it to the Stage: Reflections on Live Laptop Music from Artists"
Create Digital Music, 21 Jul 2009 Online Available <http://cdm.link/2009/07/take-it-to-the-stage-reflections-on-live-laptop-music-from-artists/>
- Magnusson, T. (2006). "Affordances and constraints in screen-based musical instruments." In *Proceedings of NordiCHI: The Nordic conference on Human-computer interaction: changing roles*, pp. 441- 444.
- Magnusson, T. (2010). "Designing Constraints: Composing and Performing with Digital Musical Systems" *Computer Music Journal*, 34 (4)
- Manning, P. D., & Manning, P. (2004). *Electronic and computer music*. New York ; Oxford University Press.
- Miranda, E.R., & M.Wanderley. (2006). *New Digital Musical Instruments: Control and Interaction beyond the Keyboard*. Middleton, WI: A-R Editions.
- Norman, D. A. (1988). *The Psychology of Everyday Things*. New York: Basic Books.
- Norman, D. A. (1999). "Affordances, Conventions, and Design." *Interactions*
- O'Modhrain, S. (2011). "A Framework for the Evaluation of Digital Musical Instruments." In *Computer Music Journal*. 35. pp. 28-42.
- PD Conception. (2013). "So Long Mishka" *pd conception - Baikonur* © 2013
- Perkis, T. & Bischoff, J. (2007). *The League of Automatic Music Composers 1978–1983*
 Notes by Tim Perkis and John Bischoff.
- Prévost. E. (1995). *No Sound is Innocent: AMM and the Practice of Self-invention, Meta-musical Narratives, Essays*. Copula
- Puckette, M.(1988)."The Patcher", *Proceedings, ICMC. San Francisco: International Computer Music Association*, pp. 420-429.

- Puckette, M. (1991). "Combining Event and Signal Processing in the MAX Graphical Programming Environment". *Computer Music Journal* 15(3) pp. 68-77.
- Puckette, M. (2002). "Max at Seventeen". *Computer Music Journal* 26 (4) pp. 31-43.
- Rule, G. (1997). "Still hacking after all these years". In *Keyboard Magazine*. April 1997. Available Online <http://www.aphextwin.nu/learn/98136121766088.shtml>
- Scaletti, C. (1990). "Composing Sound Objects in Kyma". In *Perspectives of New Music*. pp.27- 42.
- Scaletti, C. (2002). "Computer Music Languages, Kyma, and the Future". *Computer Music Journal*. 26. pp. 69-82
- Spiegel, L. (1985). "Music Mouse" Online Available <http://tamw.exxoshost.co.uk/mmouse.html>
- Svanæs, D. (1999) " Understanding Interactivity -Steps to a Phenomenology of Human Computer Interaction". Ph.D. Thesis, NTNU, Norway.
- Sutherland, R. (1994). *New Perspectives in Music*. London: Sun Tavern Fields.
- Tanaka, A.(2009). "Sensor based Musical Instruments and Interactive Music." In Dean, R. (Ed.) *Oxford Handbook of Computer Music and Digital Sound Culture*. Oxford University Press, Oxford. 2009.
- The Hub (1989). Artifact ART 1002 Liner Notes.
- Trueman, D. Cook, P., Smallwood, S., Wang, G. (2010). "PLOrk- The Princeton Laptop Orchestra, Year 1" In *Proceedings of the International Computer Music Conference*. New Orleans.
- Trueman, D. (2007). "Why a laptop orchestra?" In *Organised Sound* 12 (2), pp.171-179.
- Turner, T. (2003). "The Resonance of the Cubicle: Laptop Performance in Post-digital Musics", *Contemporary Music Review*, 22:4, pp. 81-92

- Vercoe, B. & Ellis, D. (1990). "Real-time CSOUND: Software Synthesis with Sensing and Control" *ICMC Proceedings*, Glasgow (1990), pp. 209-211
- Zadel, M. (2006). "A Software System for Laptop Performance and Improvisation". Master Thesis McGill University, Montreal, Canada.
- Zicarelli, D. (1988). "M and Jam Factory." In *Computer Music Journal* 11 MIT Press

Appendices

Appendix A

List of files included digitally

Laptop instruments

Compressed as .zip files. When a zip file is uncompressed, a main folder containing the maxproject file with all the required files and corresponding sub folders can be accessed.

- *alfabeta.zip*
- *alfabeta_redux.zip*
- *alfabeta_PolyPoly.zip*
- *at-multitouch.zip* - Includes the *at-mappingtool* abstraction.
- *at-granular.zip*
- *at-crossConv+Sampler.zip*
- *at-crossConvGRAN.zip*
- *at-noise.zip*
- *Smack-Machine.zip*

Performance examples

Audio examples of performances:

- *Ex-1_MultiTouchContPitch+at-noise.mp3*
- *Ex-2_MultiTouchGranular.mp3*
- *Ex-5_MultiTouchMappingFXs.mp3*

- *Ex-6_Keyboard_AlfabetaRedux(early_version).mp3*
- *Ex-7a_KeyboardAlfaBeta_Redux.mp3*
- *Ex-7b_KeyboardAlfaBeta_Redux.mp3*
- *Ex-8b_KeyboardPolyPoly(early_version).mp3*

Video examples of performances:

- *Ex-3_MultiTouchGranular+KeyboardPolyPoly.mp4*
- *Ex-4_MultiTouchGranularCross+FXs.mp4*
- *Ex-8a_KeyboardPolyPoly(early_version).mp4*
- *Ex-9_KeyboardPolyPoly.mp4*
- *Ex-10_SmackMachine.mp4*